Complex Network Construction for Interactive Image Segmentation Using Particle Competition and Cooperation: A New Approach

Jefferson Antonio Ribeiro Passerini and Fabricio Breve(✉)

São Paulo State University, Rio Claro, SP 13506-900, Brazil
jefferson.passerini@gmail.com, fabricio.breve@unesp.br

Abstract. In the interactive image segmentation task, the Particle Competition and Cooperation (PCC) model is fed with a complex network, which is built from the input image. In the network construction phase, a weight vector is needed to define the importance of each element in the feature set, which consists of color and location information of the corresponding pixels, thus demanding a specialist's intervention. The present paper proposes the elimination of the weight vector through modifications in the network construction phase. The proposed model and the reference model, without the use of a weight vector, were compared using 151 images extracted from the Grabcut dataset, the PASCAL VOC dataset and the Alpha matting dataset. Each model was applied 30 times to each image to obtain an error average. These simulations resulted in an error rate of only 0.49% when classifying pixels with the proposed model while the reference model had an error rate of 3.14%. The proposed method also presented less error variation in the diversity of the evaluated images, when compared to the reference model.

Keywords: Interactive image segmentation · Machine learning · Semi-supervised learning · Particle competition and cooperation

1 Introduction

Image segmentation is the process of identifying and separating relevant structures and objects from an image, for its later analysis and information extraction. This is considered one of the hardest tasks in image processing [12].

The creation of image segmentation algorithms is considered an arduous task since it depends on the specific domain of the images or image groups. An algorithm designed for a specific domain usually has poor performance when applied to multiple images from different domains. Therefore, completely automated image segmentation still remains a challenge [12].

Supported by the São Paulo Research Foundation – FAPESP (grant #2016/05669-4).

© Springer Nature Switzerland AG 2020
O. Gervasi et al. (Eds.): ICCSA 2020, LNCS 12249, pp. 935–950, 2020.
https://doi.org/10.1007/978-3-030-58799-4_67

Interactive image segmentation approaches rose as an alternative [1,13,16, 20], in which user input are used as clues to the algorithm. Many of these approaches are based on semi-supervised learning, as they are designed to handle scenarios with an abundance of unlabeled data and few labeled data, which are more onerous to be generated [10,22].

The particle competition and cooperation (PCC) model [2] is a graph-based supervised learning method. This model first converts the dataset into a non-weighted and non-orientated graph in which every data item corresponds to a node of the graph and its edges are generated from the similarity relations between the data items. The particles, which correspond to the labeled data, move in the graph cooperating with other particles of the same class and competing with particles of other classes.

Every group of particles aims to dominate the largest number of non-labeled nodes and to avoid the invasion of the labeled nodes by particles of other classes. At the end of the iterative phase, the boundaries are defined by the particles' territories.

Many graph-based semi-supervised learning models are similar and share the same characteristics of operation, globally labeling the data [22]. The particle competition and cooperation model uses a local propagation of the labels approach, so its computational cost is close to linear ($O(n)$), while other approaches have cubic computation complexity ($O(n^3)$).

The PCC model has already been implemented in some important machine learning tasks such as the detection of overlapping communities [4], learning with label noise [5], active learning [6] and interactive image segmentation [3,8,9].

When applying the PCC model to image segmentation, a complex network is build based on the image to be segmented. Each pixel is represented as a node and the labeled pixels are also represented as particles [3]. The edges are defined according to the similarity between each pair of pixels, measured by the Euclidean distance among features extracted from them, such as RGB and HSV components, and pixel localization. It is important to define a weight for each feature in this process according to their discriminative capacity in the image to be segmented. The weight vector directly affects the network construction and, therefore, it has a big impact on the PCC segmentation accuracy.

So far, the proposed methods for defining features weights work well in some images and fail in others, demonstrating the difficulty of this process. In [7], the weights are calculated with automatic methods based on the average, standard deviation and histogram of the labeled pixels. Each method was able to increase the accuracy in some tested images but failed in others. In [8], a network index is proposed to evaluate candidate networks. A genetic algorithm is used to optimize the network index, thus evolving a good network. However, the optimization process is time-consuming and the network index is based on assumptions that do not hold for all images.

This paper proposes the elimination of the weight vector. To achieve this, it changes the complex network construction in the following way: (a) a different set of features is used, (b) a new form of user annotation is introduced, (c) a new

approach is used to define the edges among network nodes, and (d) the particle influence on the network is measured before the competition process starts.

Simulations were made using 151 real-world images. 49 of them are taken from the GrabCut dataset, 99 from the PASCAL VOC dataset, and 3 images from the Alpha matting dataset. These are the same images used in [14]. The segmentation efficacy is calculated through the comparison between the segmented image using the proposed model, the reference model, and versions of the images segmented by experts and made available in the data sets.

The remaining of this paper is organized as follows: Sect. 2 describes the particle competition and cooperation model applied to image segmentation. Section 3 describes the proposed method to build the complex networks. Section 4 describes how the experiments were performed. In Sect. 5 the results are presented and discussed. Finally, in Sect. 6 we draw some conclusions.

2 Image Segmentation Using Particle Competition and Cooperation

Previous papers demonstrate the PCC model applied to image segmentation [3, 7–9]. The image is first converted into a complex network where nodes represent image pixels and edges represent the similarity between them, measured through the Euclidean distance among weighted features extracted from those pixels. The weight vector defines the importance of each feature in the set, for each processed image. Unfortunately, a different set of weights is required for each image to be processed [8,9].

Particles are created for each pixel that was labeled by the human specialist. During the iterative phase, these particles walk in the network cooperating with other particles of the same class and competing against particles from other classes for the possession of the unlabeled nodes. They use a random-greedy rule to decide the next node to visit.

Each node has a set of domination levels. Each of these levels corresponds to a class of the image. When a particle visits a node, it can increment the dominance level of its class on the node, while reducing the domination level of the other classes. Every particle has a strength level which is altered according to the nodes it visits. Every particle also has a distance table that contains the distance, measured in hops, from every visited node to its origin node. This table is dynamically updated during the particle's walk through the network.

In [7,8], a set of 23 features is extracted from the pixels: (1–2) pixel location components (line, column), (3–5) RGB components, (6–8) HSV components, (9–14) the average of the RGB and HSV components on the pixel and its neighbors, (15–20) the standard deviation of the RGB and HSV components on the pixel and its neighbors, (21–23) the ExR, ExG, and ExB components.

For the features averages and standard deviation, the 8-connected neighbors of the pixel are considered, except for the borders. All the extracted components are normalized to have a mean 0 and a standard deviation of 1. The HSV components were calculated as described by [21], and the components ExR, ExG,

and ExB were obtained from the RGB components as described in [15]. Each feature can be emphasized or not through a weight vector λ during the Euclidean distance calculation.

The network is represented by a non-directed and unweighted graph. Therefore, $G = \{V, E\}$, where $V = \{v_1, v_2, \ldots, v_n\}$ is the set of nodes and $E = \{v_i, v_j\}$ determinates the set of edges. Each node v_i corresponds to a pixel x_j. Two nodes v_i and v_j are connected if v_j is among the k-nearest neighbors of v_i, considering the Euclidean distance features of x_i and x_j, otherwise, they are disconnected.

For each node $v_i \in \{v_1, v_2, \ldots, v_L\}$ correspondent to a labeled pixel $x_i \in X_L$, a particle p_i is generated and its initial position is defined as v_i. Each particle p_j has a variable $p_j^w(t) \in [0, 1]$ to store its strength, which determines the impact generated by the particle to the visited node. The initial strength is always set to its maximum value $p_i^w(0) = 1$.

Each particle has a table of distances which is dynamically updated during the movement of a particle, in which the distance between the particle's initial position and the visited nodes are stored. It is determined by $p_j^d(t) = \{p_j^{d_1}(t), p_j^{d_2}(t), \ldots, p_j^{d_n}(t)\}$, where each element $p_j^d(t) \in [0, n-1]$ corresponds to the measured distance between the particle's original node p_j and the node v_i.

Each node v_i has a vector variable $v_i^w(t) = \{v_i^{w_1}(t), v_i^{w_2}(t), \ldots, v_i^{w_n}(t)\}$ where each element $v_i^w(t) \in [0, 1]$ represents the domination levels of the team l over the node. The sum of this vector is always a constant,

$$\sum_{l=1}^{e} v_i^{w_l} = 1. \tag{1}$$

This occurs because when a particle increases its dominance level on the node, the domination of the other groups decrease in the same proportion for all the classes, which is determined as law of conservation of forces.

The initial domination levels $v_i^w(t)$ of each node v_i is configured as:

$$W_i^{wl} = \begin{cases} 1 & \text{if } x_i \text{ is labeled and } y(x_i) = l \\ 0 & \text{if } x_i \text{ is labeled and } y(x_i) \neq l \\ \frac{1}{c} & \text{if } x_i \text{ is not labeled} \end{cases}. \tag{2}$$

During the iterations, the domain values vary according to the equation:

$$v_i^{w_l}(t+1) = \begin{cases} max\left\{0, v_i^{w_l}(t) - \frac{\Delta_v p_j^w(t)}{c-1}\right\} & \text{if } x_i \text{ is unlabeled and } l \neq p_j^f \\ v_i^{w_l}(t) + \sum_{q \neq c} v_i^{w_q}(t) - v_i^{w_q}(t+1) & \text{if } x_i \text{ is unlabeled and } l = p_j^f \\ v_i^{w_l}(t) & \text{if } x_i \text{ is labeled} \end{cases}, \tag{3}$$

where p_i^f represents the team of the particle p_j. Each particle will change the node its visiting v_i, increasing the domination level of its class $(v_i^{w_l}(t), l = p_j^f)$ on it, and decreasing the domination levels of other classes $(v_i^{w_l}(t), l \neq p_j^f)$, always respecting the law of conservation of forces.

The particle's force depends on the domination level of its team on the visited node. Therefore, in every iteration, the force of the particle $p_j^w(t)$ is updated according to the equation: $p_j^w(t+1) = v_i^{w_l}(t+1)$, where v_i is the target node and $l = p_j^f$, with l being the label of the particle's team p_j, consequently each particle p_j has its strength p_j^f configured with the domination level value of its team $w_i^{w_j}$ in the node v_i.

The model conditions the particle's force to a reducer Δ_v, in this way it is possible to control the speed in which the particles will dominate the unlabeled nodes.

When the node is visited, the distance table is updated according to the following equation:

$$p_j^{d_k}(t+1) = \begin{cases} p_j^{d_i}(t) + 1 & \text{if } p_j^{d_i}(t) + 1 < p_j^{d_k}(t) \\ p_j^{d_k}(t) & \text{otherwise} \end{cases}. \tag{4}$$

This way, when the particle moves from the current node to the target node, it checks the distance table and, if the distance from the target node is larger than the current node's distance plus 1, the table is updated. This way, it is not necessary to know the distances for every node a priori.

In every iteration, the particle must choose which node it will visit among the k-nearest neighbors from the current position, this movement is based in one of two rules: random rule or greedy rule.

In the random rule the particle randomly chooses, with equal possibilities, any of the neighboring vertices from the position in which the particle is located. This rule does not take into account the levels of domain or distances of the original node, being useful for exploration and acquisition of new nodes. It is defined by the equation:

$$p(v_i|p_j) = \frac{W_{qi}}{\sum_{\mu=1}^{n} W_{q\mu}}, \tag{5}$$

where q is the node's index where the current particle p_j is located, therefore $W_{qi} = 1$ if there is an edge between the current node and any other node v_i or else $W_{qi} = 0$.

In the greedy rule, the particle randomly chooses any one of the neighboring nodes from the current node, with probabilities calculated directly proportional to the dominance level of this particle's team in every neighboring node, and inversely proportional to the distance of these neighbors to the particle's local node. This movement is useful for the team's territory defense. It is defined by the following equation:

$$p(v_i|p_j) = \frac{W_{qi} v_i^{w_l} \left(1 + p_j^{d_i}\right)^{-2}}{\sum_{\mu=1}^{n} W_{q\mu} v_\mu^{\omega_l} \left(1 + p_j^{d_\mu}\right)^{-2}}, \tag{6}$$

where q is the node's index where the current particle p_j is located, and $l = p_j^f$, with p_j^f being the class label of the particle p_j.

At each iteration, every particle has the probability P_{grd} of choosing the greedy movement and the probability $1-P_{grd}$ of choosing the random movement, with $0 \leq p_{grd} \leq 1$.

The algorithm stops when it reaches the maximum number of iterations ($maxIte$), or when the node's domination levels reach some stability. The stability is inferred when, on average, the nodes had improved below a threshold ($controlStop$) in their maximum domination level in a given iteration interval ($maxStop$).

3 Constructing a New Complex Network for the Model

Based on what has been presented in previous papers, this paper proposes a change in how the complex network is built and the elimination of the weight vector that defines the importance of each feature within the set.

The original model receives the image to be segmented (for example: Fig. 1a) and the user input (Fig. 1b). In this new approach, it is also possible to delimit, in the image, the region of interest where the object to be segmented is found (Fig. 1c), to reduce the processing scope.

Figure 1d displays a composition of the algorithm's input information. The cut polygon represented in Fig. 1c is optional to the model because there can be different situations that the cut in the original image is not of interest since the segmentation object of choice is set across most of the image. Regardless of the usage of the cut polygon by the algorithm or not, the complex network assembly process is the same.

The input image passes through a bicubic interpolation process to reduce its dimension (amount of pixels) allowing the processing of larger images. Later, the bilinear interpolation algorithm is used for information recomposition after the segmentation processing, as described in [11,18,19].

Once the image is reduced, the extraction of the image's pixels features set is started. This new proposed approach uses less features than its predecessors: (1–2) pixel location components (line, column), (3–5) RGB components, (6) only the V (value) component of the HSV system [21], (7–9) the color components ExR, ExG, ExB defined in [15], and (10) a new feature extracted using Otsu's binarization algorithm [17]. Figure 2 shows graphic representations of each color feature extracted from an image from the Grabcut data set.

The features are normalized to have an average of 0 and a standard deviation of 1. Thereafter occurs the non-oriented graph generation, which will represent the pixels of the image through its nodes, while the edges represent the similarity relationships between the pixels. As has been previously stated, the similarity relationships are determined by measuring the Euclidean distance among the extracted features from the image to be segmented.

In the reference model, each node is connected to its k-nearest neighbors, considering the Euclidean distance among pixel features, and k is a parameter set by the user. In the proposed model, k is fixed, and each node is connected to its 192 nearest neighbors. Another 8 connections are made based in the pixel

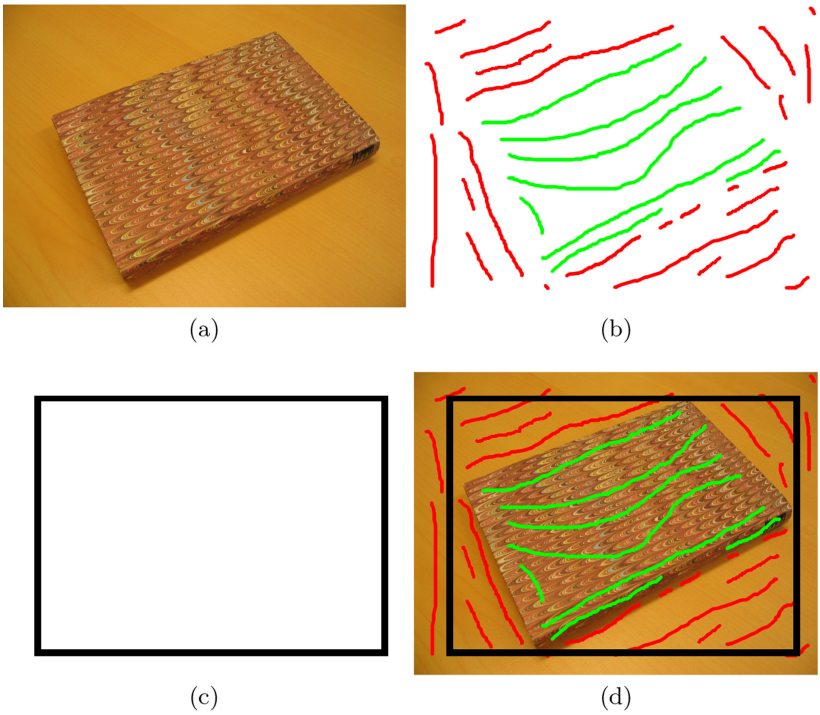

Fig. 1. Input information: (a) the real-world images to be segmented, (b) the "scribbles" provided by the user, (c) the cut polygon, and (d) overlay image for visualization only.

spatial neighborhood, defined by a 3×3 window, i.e., the corresponding node will be linked to the nodes corresponding to its 8 physically adjacent pixels. This change was introduced to consider the pixel physical neighborhood when creating a complex network since the original approach cannot guarantee they will be connected.

Another alteration is that the proposed model verifies whether two linked nodes are of different classes and, if this happens, the new edge will not be generated. Nodes can only be linked with nodes of the same class or with unlabeled nodes.

Once the complex network is defined, this approach considers the influence of a previously labeled pixel (particle) in relation to its physical neighbors in the original image. This alteration also aims to reinforce the relationship information among the image's pixels before the iterative phase, but now in relation to the pixels labeled by the user in a 5×5 window.

It is considered that a particle (labeled pixel) influences other nodes in the network up to the distance of 2 hops in the structure. It must be emphasized that, in this approach, the distance also takes into consideration the spatial distance of the pixel present in the image. This alteration aims to value the

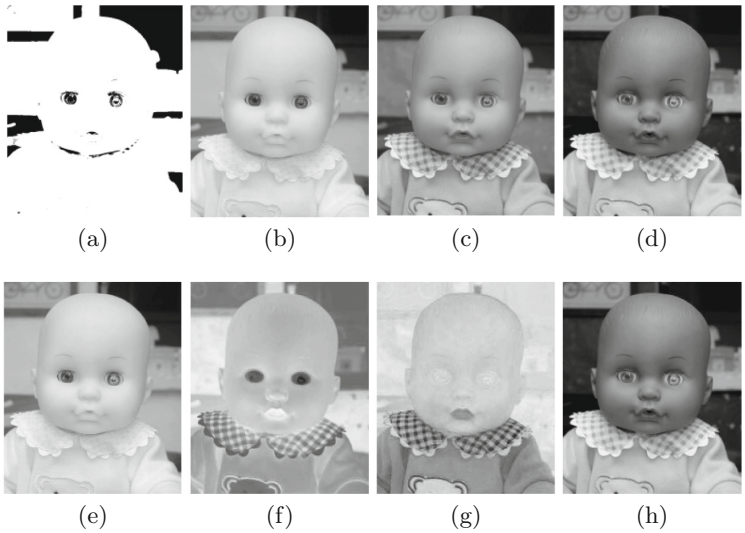

Fig. 2. Proposed algorithm's extracted features representation: (a) the binarization made with Otsu's algorithm, (b–d) the RGB components, respectively, (e) the V component from the HSV system, (f–h) the ExR, ExG, and ExB components, respectively.

pixel's relationships of the spatial neighborhood in the image to be analyzed since a neighboring unlabeled pixel tends to be part of the same class.

Therefore, unlabeled nodes which are 1 hop away from a labeled node will have an increment of 0.2 in its domination vector, referent to the nearby particle's class. There is the possibility that the unlabeled node suffers the influence in its domination vector of more than one nearby particle. Unlabeled nodes which are 2 hops away from a labeled node suffer less influence in their domination vector. The increment is only 0.1 in the nearby particle's class. This process is illustrated in Fig. 3.

This approach respects the law of conservation of forces, previously described. If a pixel has its domination vector completely dominated by a given team of particles, it will still be subject to changes during the iterative phase, differently from the labeled nodes, which have fixed domination levels.

After the network construction and initial settings with the proposed approach, the iterative phase takes place exactly as described in earlier particle competition and cooperation models.

4 Experiments

Two models were implemented to perform the tests with the proposed approach in this paper. The first one is the reference model [3,7–9], with the 23 features already described. The weight vector λ was defined so all the features had the

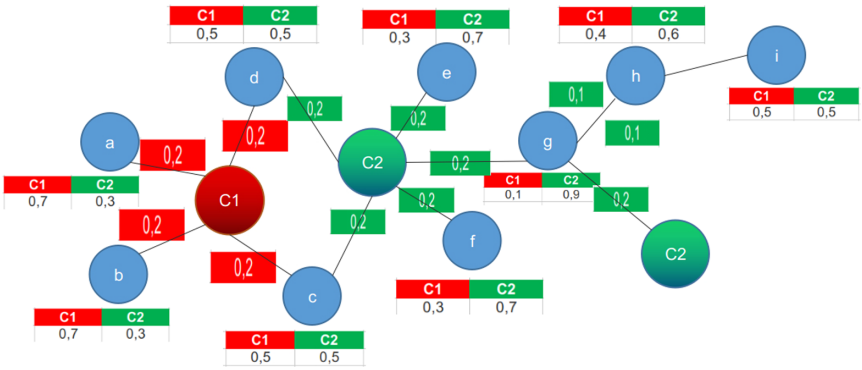

Fig. 3. Representation of the labeled pixels (particles) influence on the spatial neighborhood of the original image.

same weight. The reference model receives the interpolation approach as an increment in its execution to reduce the size of the images to be processed. The second is the proposed model, as described in the previous session.

151 real-world images taken from the GrabCut dataset, the PASCAL VOC dataset, and from the Alpha matting dataset are used to evaluate both models. These are the same images used in [14]. The markings (labels) defined for the tests and the cut polygons used in this work are available at Github[1]. In our simulations, each image was processed 30 times by each model to calculate the models' average performance. In total 4,530 executions were performed with each algorithm.

The error rate is determined by comparing the algorithms' output with the ground-truth images. Notice that the ground-truth images have some pixels (in grey) representing divergence in classification among the specialists. Those pixels were not considered in the error calculation.

Both models were configured to use 200 neighbors, $\Delta_v = 0.1$, $P_{grd} = 0.5$, $maxIte = 1,000,000$, $maxStop = 15,000$, and $controlStop = 0.001$. The same images and annotated pixels were used for both models.

The simulations were performed in an Asus laptop model S46C configured with a Quadcore 2 Ghz Intel i7 processor, model 3537U, with 16 GB of RAM, running the Linux Deepin 15.5 Desktop operating system. The models' development was made using the Python version 2.7 programming language.

5 Results and Discussion

The experiments were performed as discussed in Sect. 4. It was observed that in five images from the data set it was not possible to apply the cut polygon feature,

[1] Available at https://github.com/jeffersonarpasserini/dataset-interactive-algorithms.git.

due to the subject being segmented occupying most of the original image. Those images are presented in Table 1 with the error rates achieved by both models.

Table 1. Error rates in the five images that did not use the cut polygon resource, as achieved by the proposed model and the reference model.

Image name	Proposed	Reference
Baby_2007_006647	1.17%	4.57%
cross	0.48%	1.79%
gt02	0.52%	1.27%
gt07	0.21%	0.64%
gt13	1.08%	2.11%
Average	0.64%	1.72%

The proposed method obtained the best pixel classification accuracy in comparison to the reference method. In the average, the proposed method obtained an error rate of 0.64% while the reference method obtained an error rate of 1.72%. The processed images can be observed in Fig. 4.

Regarding processing time, the reference method took 1371.08 s to output the results presented in Fig. 4 while the proposed method took 1205.06 s.

Table 2 shows the five data set images in which the proposed method obtained the lowest error rates. In this case, all the images used the cut polygon feature during the proposed model execution to cut the segmentation area of interest, whilst in the reference method, the entire image was used. This cut process used in the images resulted in smaller networks for the proposed method in comparison to the reference method. The average of the images' complex network nodes, described in Table 2, was of 2,089 nodes with 554,455 edges in the proposed method whilst, for the reference method, it was of 4,592 nodes and 2,102,821 edges.

Table 2. Error rates in the images with the lowest error rates achieved by the proposed method.

Image name	Proposed	Reference
Monitor_2007_003011	0.02%	1.09%
Train_2007_004627	0.09%	0.76%
Car_2008_001716	0.10%	2.51%
Monitor_2007_004193	0.11%	3.00%
Person_2007_002639	0.12%	2.47%
Average	0.08%	1.94%

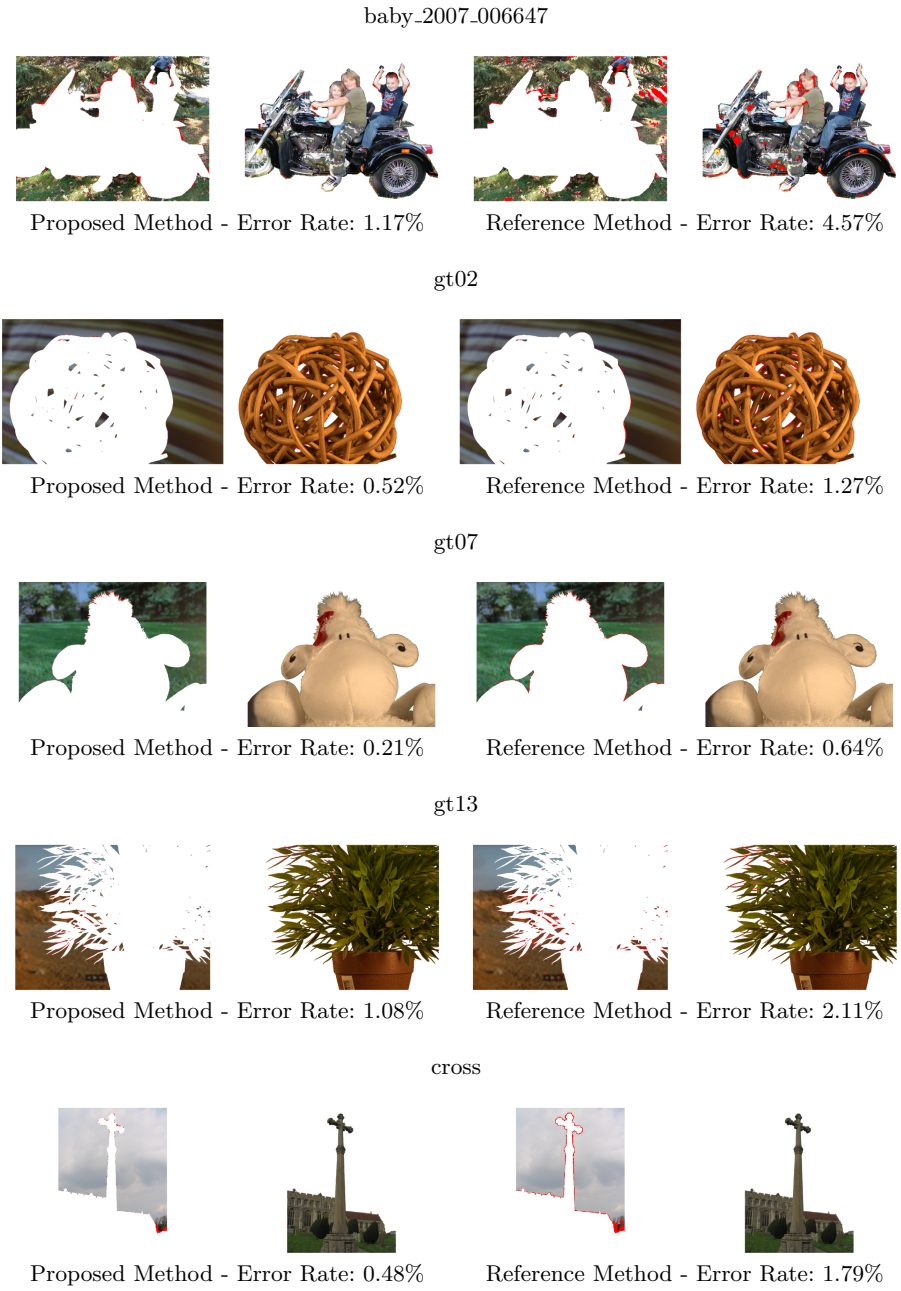

Fig. 4. Image segmentation results by the proposed model and the reference model without the use of the cut polygon tool. Misclassified pixels are highlighted in red. (Color figure online)

By analyzing Table 2, it can be verified that the proposed method, with a 0.08% error rate, obtained a superior performance when compared to the reference method, which obtained an error rate of 1.94%. Regarding the average execution time, the proposed method performed the images' segmentation in 255.46 s whilst the reference method performed it in 823.30 s. The higher execution time difference is justified by the difference in the complex network generation which defined smaller networks for the proposed method. These processed images can be observed in Fig. 5.

The general performance of both methods, considering the processing of all the 151 images, is shown in Table 3 and Table 4. In Table 3, the size of the generated networks, for each method, can be verified. The proposed method generated smaller networks, in both nodes and edges, in comparison to the reference method. The 151 images studied had an average of 200,124 pixels, from which 2,783 are unlabeled. The proposed method generated complex networks with an average of 7,538 nodes and 838,564 edges, whilst the reference method generated complex networks with a higher average of 17,946 nodes and 2,354,555 edges.

Table 3. Generated complex networks average characteristics by the proposed model and the reference model.

	# Pixels		Characteristics		
Method	All	Unlabeled	Particles	Nodes	Edges
Proposed	200,124	2,783	2,860	7,538	838,564
Reference	200,124	2,783	5,487	17,946	2,354,555

Table 4 shows the average error rate when classifying the pixels for all the 151 images studied, using both the proposed method and the reference method. The proposed method achieved an average error rate of 0.49% while the reference method achieved 3.14%. Regarding execution time, the proposed method was beneficiated by the generation of smaller networks. It took an average of 432.54 s to segment the images, whilst the reference method performed the task in an average time of 1082.94 s.

Table 4. Average error rate and execution time in the 151 images by the proposed method and the reference method.

Method	Error rate	Time(s)
Proposed	0.49%	432.54
Reference	3.14%	1082.94

Figure 6 shows the relationship between the error rate and the execution time to perform the segmentation of each of the 151 images. The proposed method has

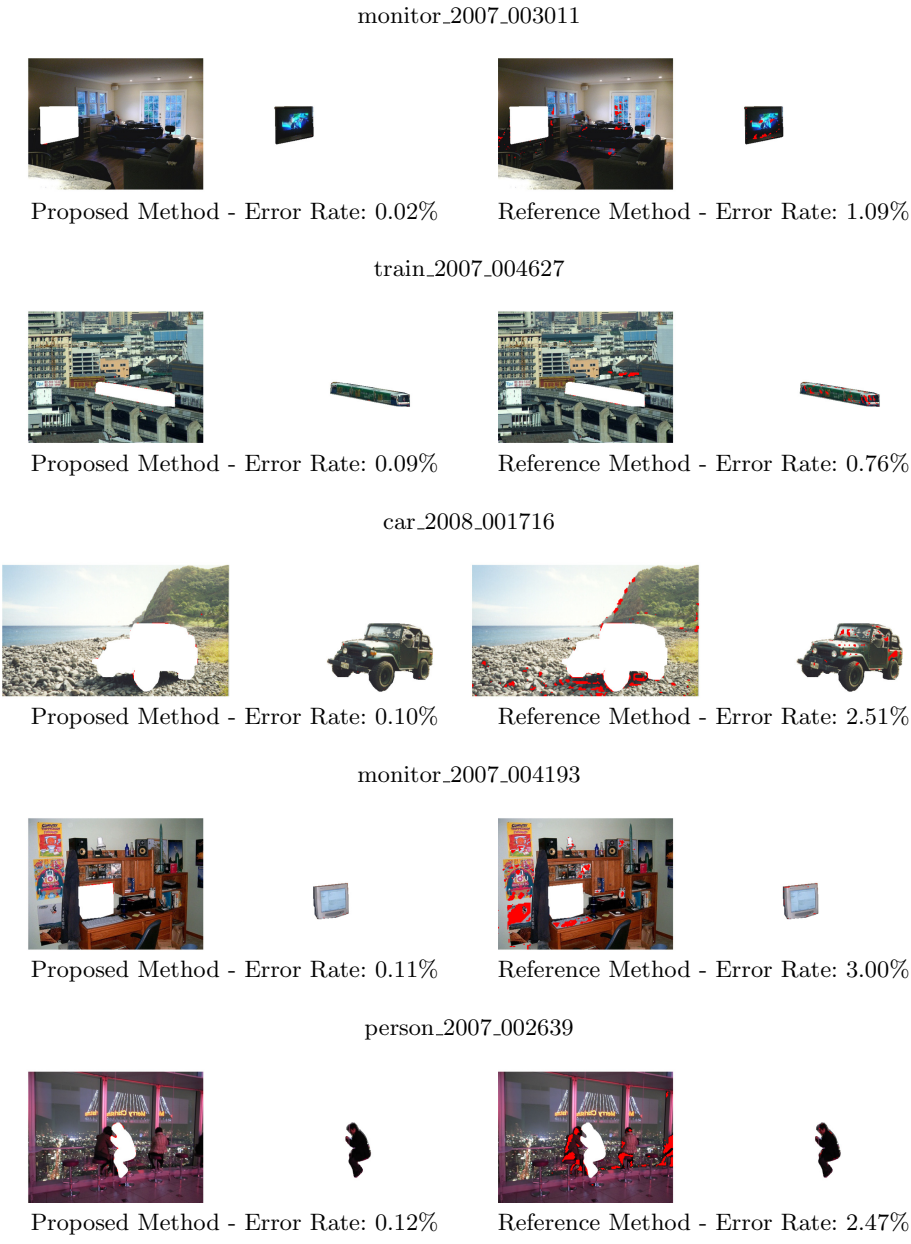

Fig. 5. Images segmentation results by the proposed model with the use of the cut polygon tool and the reference model. Misclassified pixels are highlighted in red. (Color figure online)

better stability in comparison to the reference method. The images processed by the proposed method are concentrated in a specific region of the graphic, whilst the reference method image's processing results are spread through the graphic. Therefore, the proposed method is less susceptible to features variations among the different images presented to it in this study.

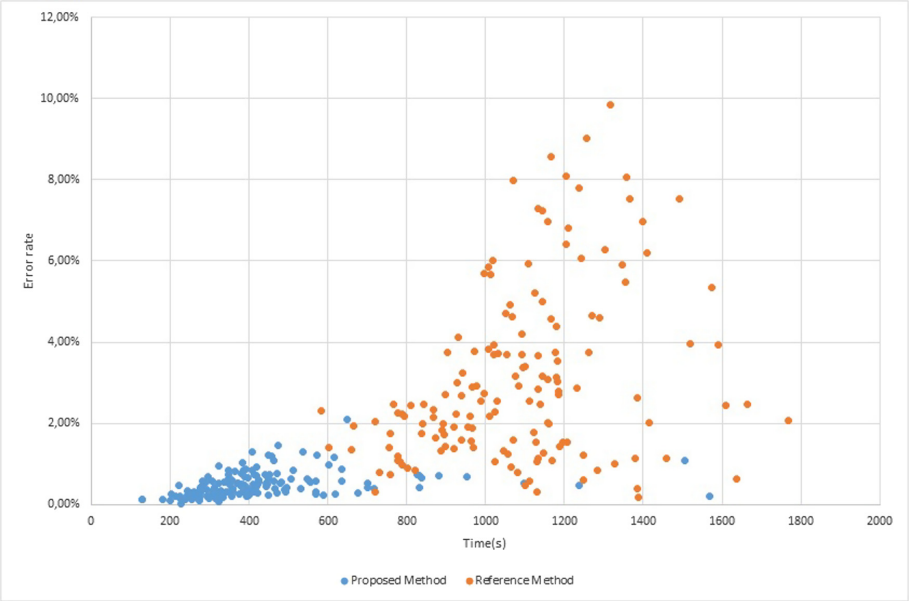

Fig. 6. Relation Analysis between error rate and processing time presented by the proposed model and reference model.

6 Conclusions

Autonomous segmentation represents one of the most complex image processing tasks since there is not an image segmentation algorithm that can be used in every situation. This paper presented a methodology to improve the automation level of the particle competition and cooperation model for image segmentation.

The proposed approach eliminates the weight vector λ from the model since it is a parameter that needed to be set for each image to be segmented, requiring some expertise of the user. This elimination increased the model automation and it is achieved by optimizing the network construction phase, without changes in the particle competition and cooperation iterative phase.

The computer simulations have shown that the proposed method achieved better results when compared to the reference method, both in accuracy and performance. The proposed method obtained an average error rate of only 0.49%, whilst the reference method obtained an average error rate of 3.14% for the 151 images studied.

The processing time was also reduced. The proposed method performed the segmentation in an average time of 432.54 s, whilst the reference method took an average time of 1082.94 s. This fact is justified by the average size of the networks generated by the studied models. The proposed method generated networks significantly smaller than the reference method.

References

1. Boykov, Y., Jolly, M.P.: Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images. In: Eighth IEEE International Conference on Computer Vision, 2001. ICCV 2001. Proceedings, vol. 1, pp. 105–112 (2001)
2. Breve, F., Zhao, L., Quiles, M., Liu, J., Pedrycz, W.: Particle competition and cooperation in networks for semi-supervised learning. Knowl. Data Eng. **24**(9), 1686–1698 (2012)
3. Breve, F.A., Quiles, M.G., Zhao, L.: Interactive image segmentation using particle competition and cooperation In: Proceedings of the International Joint Conference on Neural Networks (IJCNN2015), pp. 1–8. Killarney, Irlanda (2015). https://dx.doi.org/10.1109/IJCNN.2015.7280570
4. Breve, F., Zhao, L.: Fuzzy community structure detection by particle competition and cooperation. Soft Comput. (2013). https://doi.org/10.1007/s00500-012-0924-3
5. Breve, F.A., Zhao, L., Quiles, M.G.: Particle competition and cooperation for semi-supervised learning with label noise. Neurocomputing (2015). https://doi.org/10.1016/j.neucom.2014.08.082
6. Breve, F.A.: Active semi-supervised learning using particle competition and cooperation in networks. In: Neural Networks (IJCNN) (2013)
7. Breve, F.A.: Auto feature weight for interactive image segmentation using particle competition and cooperation. In: Proceedings - XI Workshop de Visão Computacional WVC2015, pp. 164–169. XI Workshop de Visão Computacional (WVC2015), October 2015
8. Breve, F.: Building networks for image segmentation using particle competition and cooperation. In: Gervasi, O., et al. (eds.) ICCSA 2017. LNCS, vol. 10404, pp. 217–231. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-62392-4_16
9. Breve, F.A., Quiles, M.G., Zhao, L.: Interactive image segmentation of non-contiguous classes using particle competition and cooperation. In: Gervasi O. et al. (eds.) Computational Science and Its Applications – ICCSA 2015. ICCSA 2015. LNCS, vol. 9155. Springer, Cham. https://doi.org/10.1007/978-3-319-21404-7_15
10. Chapelle, O., Schölkopf, B., Zien, A.: Semi-Supervised Learning. Adaptive Computation and Machine Learning. MIT Press, Cambridge (2006)
11. Getreuer, P.: Linear methods for image interpolation. Image Process. On Line **1**, 238–259 (2011)
12. Gonzalez, R.C., Woods, R.E.: Processamento Digital de Imagens, 3 edn. Longman do Brasil (2010)
13. Grady, L.: Random walks for image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **28**(11), 1768–1783 (2006). https://doi.org/10.1109/TPAMI.2006.233
14. Gulshan, V., Rother, C., Criminisi, A., Blake, A., Zisserman, A.: Geodesic star convexity for interactive image segmentation. In: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 3129–3136. IEEE (2010)

15. Lichman, M.: UCI machine learning repository (2013). http://archive.ics.uci.edu/ml/datasets/image+segmentation
16. Malmberg, F.: Graph-based methods for interactive image segmentation. Uppsala University, Centre for Image Analysis, Computerized Image Analysis and Human Computer Interaction (2011)
17. Otsu, N.: A threshold selection method from gray-level histograms. IEEE Trans. Syst. Man Cybern. **9**, 62–66 (1979)
18. Pedrini, H., Schwartz, W.R.: Análise de Imagens Digitais: Princípios. Algoritmos e Aplicações. Thomson Learning, São Paulo (2007)
19. Prajapati, A., Naik, S., Mehta, S.: Evaluation of different image interpolation algorithms. Int. J. Comput. Appl. **58**(12), 6–12 (2012)
20. Rother, C., Kolmogorov, V., Blake, A.: "GrabCut": interactive foreground extraction using iterated graph cuts. ACM Trans. Graph. **23**(3), 309–314 (2004). https://doi.org/10.1145/1015706.1015720
21. Smith, A.R.: Color gamut transform pairs. ACM SIGGRAPH Comput. Graph. (1978). https://doi.org/10.1145/965139.807361
22. Zhu, X.: Semi-supervised learning literature survey. Technical report 1530, Computer Sciences, University of Wisconsin-Madison (2005)