Evolved Cellular Automata for Edge Detection

Alina Enescu
aenescu@cs.ubbcluj.ro
Babes–Bolyai University
Cluj–Napoca, Romania

Anca Andreica
anca@cs.ubbcluj.ro
Babes–Bolyai University
Cluj–Napoca, Romania

Laura Diosan
lauras@cs.ubbcluj.ro
Babes–Bolyai University
Cluj–Napoca, Romania

ABSTRACT

Cellular Automata (CA) can be successfully applied in various image processing tasks because they have a number of advantages over the traditional methods of computations: simplicity of implementation, the complexity of behaviour, parallelisation, extensibility, scalability, robustness. In this paper, an edge detection method for binary images, based on CA and Evolutionary Algorithms (EA) is presented. The rule of a two-dimensional CA is evolved by the means of two EAs, one that evolves the rule to detect edge points and another one that evolves the rule to detect non-edge points. The focus is on the implementation of the EAs, how individuals are represented, how the evolving process is evaluated and which genetic operators are used. The results of the experiments show a better performance of the proposed approach in comparison with similar approaches presented in the literature.

CCS CONCEPTS

• **Computing methodologies → Interest point and salient region detections**.

KEYWORDS

Edge detection, Cellular Automata, Evolutionary Computation

ACM Reference Format:
Alina Enescu, Anca Andreica, and Laura Diosan. 2019. Evolved Cellular Automata for Edge Detection. In *Genetic and Evolutionary Computation Conference Companion (GECCO '19 Companion), July 13–17, 2019, Prague, Czech Republic.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3319619.3322043

1 INTRODUCTION

Processing images is a problem known for a long time. One such image processing task is the edge detection which resumes to finding the boundary of an object. The boundary occurs when significantly local changes are detected [2]. A local change may refer to the difference of intensity of a pixel in contrast with some of its neighbours. Edge detection could be used as a step in a more complex process such as object recognition, object localization or many others. A potential approach to implement an edge detector is by using Cellular Automata (CA). CA is a simple model of parallel computation in which a cell changes its state based on its neighbours' states and its own state, according to a transition rule ρ. CAs have been considered for a series of applications such as cryptography, traffic simulation, image processing and many others [6, 7, 9]. The use of CA in edge detection brings several advantages such as ease of implementation, parallel implementation, possibility to work with images of different types (binary, grey or colour) and of different sizes (2D or 3D).

The goal of this paper is to validate once more the potential of CA as an edge detector and to achieve a better performance of CA. For this purpose, the CA's rule is optimized to detect the edges, with the aid of an Evolutionary Algorithm (EA). The evolved CA is evaluated by the means of comparison with (human) ground truth, and compared with similar approaches existing in the literature [1, 3, 8].

This paper is divided as follow: firstly, it is presented in detail the proposed approach, then the performed experiments and the obtained results are shown, ending with the conclusion and future work.

2 PROPOSED APPROACH

The proposed approach, similar to [1, 3, 8] optimizes the CA's rule by the means of an EA. As in [1, 3, 8] an EA chromosome encodes the CA's rule and is represented as a packet of pairs: the neighbourhood's state and the new value. A specific type of these pairs are so called linear rules[1]. Our EA performs a horizontal crossover (in [1] the vertical crossover is also used), followed by a bit flip mutation. Unlike [1, 3, 8], the fitness function corresponds to Dice similarity coefficient and the elitist sketch of EA is actually involved in the proposed approach.

The proposed approach has a binary image as input, the initial image on which the training is performed. The first step is to extract the CA initial state from the input image. Each CA's cell corresponds to a pixel and has two components: the state which is represented by the pixel's value and the eight neighbours which is a list of pairs (value, position) [2].

The next step is running the two EAs, one that trains the rules to detect edge points, another one that trains the rules to detect non–edge points. For the first EA, the initial population is generated as follow: n packets of pairs, having only the new value 0, are generated for each chromosome. The chromosomes are evaluated and the best one is kept to be added to the new population. Selection, crossover and mutation are applied to the current population and the offspring are added to the new population. The process is

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
GECCO '19 Companion, July 13–17, 2019, Prague, Czech Republic
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-6748-6/19/07…$15.00
https://doi.org/10.1145/3319619.3322043

[1] a linear rule is actually the arithmetic sum of fundamentals rules [7]

[2] The position of each neighbour is given by: $\begin{matrix} 64 & 128 & 256 \\ 32 & 1 & 2 \\ 16 & 8 & 4 \end{matrix}$, where the right neighbour position has value 2, the left has value 32 and so on.

repeated for G generations. Then the second EA is executed with the same configuration, except for the new value of each pair from the CA rule which is 1. When both EAs end, the corresponding best chromosomes encode the rule for two CA:

- the CA which involves the best packet of pairs trained to detect edge points called *Edge–CA* and
- the CA which involves the best packet of pairs trained to detect non–edge points called *NonEdge–CA*.

To these two, another two CAs are added:

- the CA which involves the packet obtained by concatenating the pairs of both best packets called *Mixed–CA*, with the mention that if two identical linear rules but with different central cell's new state exist, the one with central cell's new state 0 will be applied, and
- the CA which involves the packet obtained by reunion of the pairs of both best packets called *FilterMixed–CA*, with the mention that in cases in which two identical linear rules but with different central cell's new state exist, both of them are removed.

3 NUMERICAL RESULTS

The training dataset used for experiments consists of 100 images of size 100×100 from the ones provided in [4]. Each one of the 100 images used for training is given as input to the two EAs, the one that trains rules to detect edge points, and the one that trains rules to detect non–edge points. Each EA has a population of 50 chromosomes as packets containing a variable number of pairs $n \in [10, 20]$, and each of them runs for 50 generations. For each image, 10 individually runs are performed.

First of all, we want to investigate which of the four CAs has the best potential for edge detection. The first experiment consists in training the two EAs for all images of the considered collection. Four output images (with detected edges) can be obtained (starting from a given image) by applying the four evolved CA (*Edge–CA, NonEdge–CA, Mixed–CA, FilterMixed–CA*): *Best edge trained rules image*, *Best non–edge trained rules image*, *Unfiltered best rules image* and *Filtered best rules image*. To compare the edge detection quality of the four CAs, we computed the performance measures: Mean Squared Error (MSE) and Peak Signal to Noise Rate (PSNR) [5]. Firstly, on all resulted images from the four categories, the MSE and PSNR were computed, and the mean of MSE values, respectively PSNR values are **0.223664185**, **0.2013**, **0.2117**, **0.185**, respectively **7.2633**, **7.5143**, **7.5901**, **7.9452**[3]. The smallest value of MSE, respectively the greatest value of PSNR is obtained for FilterMixed–CA, one can conclude that FilterMixed–CA performs better than the other CA.

Secondly, we want to compare the performance of the proposed approach relative to the other similar approaches found in the literature. We are interested to have the same perspective over the compared training methods: $G = 50$ generations were chosen for all the experiments, the population of each training algorithm [1, 3, 8] has $N = 50$ packets of $n = 15$ rules and 10 individually runs on each image are performed. Performance of the Filtered–CA, identified as the best in the previous experiment, is compared with the performance of methods presented in [1, 3, 8]. The mean of the performance values over the 1000 images obtained from each method was computed: the values of MSE are **0.185**, **0.305**, **0.2227**, **0.2264** and the values of PSNR are **7.9452**, **5.8519**, **7.3048**, **7.0579**[4]. As it can be deduced from the obtained values of MSE and PSNR respectively, the proposed CA performs better than all the aforementioned methods [1, 3, 8].

4 CONCLUSIONS

We have presented an edge detection method for binary images based on Evolved Cellular Automata. Once more, it was shown the compatibility and the capability of CA in image processing tasks, in this case, the edge detection. Based on experiments shown in this paper, it can be said that our approach performs better than those already proposed in the specialised literature [1, 3, 8].

This method distinguishes from previous methods by dividing the edge detection problem into edge detection sub–problem and non-âĂŞedge detection sub–problem. Additionally, it distinguishes from the previous methods by choosing a varying number of rules in the packets of rules. A similar technique could be used in the implementation of a CA based method for grey images.

REFERENCES

[1] Mohamed Batouche, Souham Meshoul, and Ali Abbassene. 2006. On solving edge detection by emergence. In *International Conference on Industrial, Engineering and other Applications of Applied Intelligent Systems*. Springer, 800–808.
[2] R. Jain, R. Kasturi, and B. G. Schunck. 1995. Edge detection. In *Machine vision*, Vol. 5. McGraw-Hill New York, 140–185.
[3] O. Kazar and S. Slatnia. 2011. Evolutionary Cellular Automata for Image Segmentation and Noise Filtering Using Genetic Algorithms. *Journal of Applied Computer Science & Mathematics* 11 (2011).
[4] D. Martin, C. Fowlkes, D. Tal, and J. Malik. 2001. A Database of Human Segmented Natural Images and its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics. In *Proc. 8th Int'l Conf. Computer Vision*, Vol. 2. 416–423.
[5] M. H. Mofrad, S. Sadeghi, A. Rezvanian, and M. R. Meybodi. 2015. Cellular edge detection: Combining cellular automata and cellular learning automata. *AEU-International Journal of Electronics and Communications* 69, 9 (2015), 1282–1290.
[6] K. Nagel and M. Schreckenberg. 1992. A cellular automaton model for freeway traffic. *Journal de physique I* 2, 12 (1992), 2221–2229.
[7] P. L. Rosin and X. Sun. 2014. Edge Detection Using Cellular Automata. In *Cellular automata in image processing and geometry*. Springer, 85–103.
[8] Sihem Slatnia, Mohamed Batouche, and Kamal E Melkemi. 2007. Evolutionary cellular automata based-approach for edge detection. In *International Workshop on Fuzzy Logic and Applications*. Springer, 404–411.
[9] S. Wolfram. 1985. Cryptography with cellular automata. In *Conference on the Theory and Application of Cryptographic Techniques*. Springer, 429–432.

[3] The values are given in this order: Edge–CA, NonEdge–CA, Mixed–CA, FilterMixed–CA

[4] The values are given in this order: FilterMixed–CA, [1, 3, 8]