ISSN 2278-3091
Volume 9, No.3, May - June 2020
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse69932020.pdf
https://doi.org/10.30534/ijatcse/2020/69932020

Object Detection: Optimization of training Data in Genetic Programming

Viddulata. A. Patil
SVERI's College of Engineering, Pandharpur, viddulata@gmail.com

ABSTRACT

In this paper new approach for the optimization of training data for object detection particularly object localization problems in genetic programming (GP) is discussed. For the fitness function, the weighted F-measure of a genetic program is used considering localization fitness values of the detected object locations. The training data is categorized into four types: *exact center*, *close to center*, *include center*, and *background* for investigation data with this fitness function. An existing fitness function using above approach is examined and compared on three object detection problems of increasing difficulty. The results also suggest that using different proportion of all these data types to optimal results can be achieved which reduces time required for detection of objects. Only two data types Exact center type and close to center type data can be used to produce good detection results. As background type data contains less information we can neglect that type of data.

Key words : Evolutionary computing, fitness function, Genetic programming, object detection, object localization, object recognition, object classification, training data

1. INTRODUCTION

As most of the images are captured in electronic form, the need for of image processing is increasing. *I*n order to get an enhanced image or to extract some useful information from it, Image processing is a method used to perform some operations on image. Finding objects of interest in database of images is one of the applications of image processing. Thus object recognition and classification tasks arise in a very wide variety of practical situations, such as detecting faces from video images, finding tanks and helicopters from satellite images, identifying suspected terrorists from fingerprint images, and diagnosing medical conditions from X-rays. In many cases, people (possibly highly trained experts) perform the recognition/ classification tasks well, but there is either a shortage of such experts or the cost of people is too high. Therefore if the amount of image data containing objects of interest that need to be classified and recognized is given, then automatic computer based classification and recognition programs / systems are of immense social and economic value [1], [2].

Genetic programming (GP) is a relatively recent and fast developing approach to automatic programming. In GP, solutions to a problem can be represented in different forms but are usually interpreted as computer programs. Darwinian principles of natural selection and recombination are used to evolve a population of programs towards an effective solution to specific problems. The flexibility and expressiveness of computer program representation, which combined with the powerful capabilities of evolutionary search, make GP an exciting new way to solve a great variety of problems.

Various fitness functions have been devised for object detection, with varying success [3, 4, 5, 9, 11, 12, 13, 16, 17]. These tend to combine many parameters using scaling factors which specify the relative importance of each parameter, with no obvious indication of what scaling factors are good for a given problem. Many of these fitness functions for localization require clustering to be performed to group multiple localizations of single objects into a single point before the fitness is determined [14, 13, 12, 15].

Organizing training data is critical to any learning approaches. The previous approaches in object detection tend to use all possible positions of the large image in training an object detector. However, this usually requires a very long training time due to the use of a large number of positions on the background.

This paper aims to investigate a novel approach to optimize the training data in GP for object detection, in particular localization to reduce the training time.

2. METHODOLOGY

Object detection is the task of processing an image to both localize a particular object or objects and to then classify each object found. The process for object detection is shown in Figure 1.

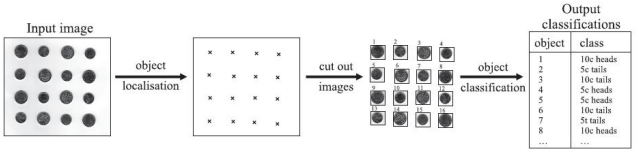


Figure 1: An overview of the Object Detection Process

A trained localizer applied to raw image, producing a set of points found to be the positions of these objects. Single objects could have multiple positions ("localizations"), however ideally there would be exactly one localization per object. Regions of the image are then "cut out" at each of the positions specified. Each of these cutouts is then classified using the trained classifier. This method treats all objects of multiple classes as a single "object of interest" class for the purpose of localization, and the classification stage handles attaching correct class labels. The object localization stage is performed by means of a window which sweeps over the whole image, and for each position extracts the features and passes them to the trained localizer. The localizer then determines whether each position is an object or not (i.e. background).

The object localization stage is performed by means of a window sweeping method which sweeps over the whole image and for each position extracts the features and passes them to the trained localizer. Figure 2 shows how sweeping window sweeps on image. The localizer then determines whether each position is an object or not (i.e. background). The window-sweeping method is a dimensionality reduction technique used to extract image features such as pixel statistics from the entire image. The window-sweeping method involves moving a fixed-size input window across an image, pixel by pixel, extracting image features at every location of the sweeping window. The size of the input window is usually large enough to contain the largest object of interest in the image but also small enough as not so miss out on too much detail when extracting features.

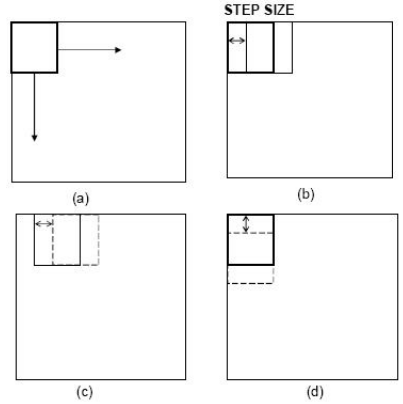


Figure 2: Figure (a) Original position of sweeping window, (b) Sweeping window at first step along the x-axis, (c) Sweeping window at second step along x-axis and (d) Sweeping window at first step along y-axis

We used the fitness function based on a "Relative Localization Weighted F-measure" (RLWF) [3]. This fitness function attempts to recognize the goodness of individual localizations made by the genetic program. Each localization is allocated a weight instead of using either correct or incorrect to represent localization which is called as the localization fitness, LF. This represents its individual worth and counts towards the overall fitness. According to the relative location, or the distance of the localization from the center of the closest object Each weight is calculated , as shown in Equation 1.

$$LF(x,y)=\begin{cases} 1-\frac{\sqrt{x^2+y^2}}{r}, & \text{if } \sqrt{x^2+y^2}\le r \\ 0, & otherwise \end{cases} \quad (1)$$

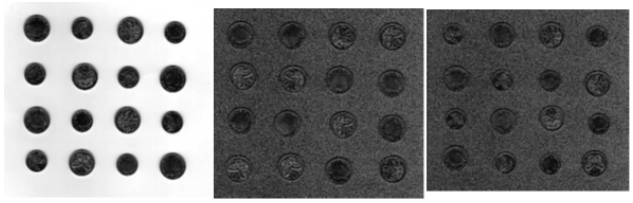
where $\sqrt{x^2+y^2}$ is the distance of the localization position (x, y) from target object centre, and r is called the "localization fitness radius", defined by the user. In this system, r is set to the radius of the largest object. We used the localization fitness to calculate this fitness function, as shown in Eq.s 2 to 4. The precision and recall are calculated by taking the localization fitness for all the localizations of each object and dividing this by the total number of localizations or total number of target objects respectively.

$$WP = \frac{\sum_{i=1}^{N}\sum_{j=1}^{Li} LF(xij,yij)}{\sum_{i=1}^{N} Li} \quad (2)$$

$$WR = \frac{\sum_{i=1}^{N}\frac{\sum_{j=1}^{Li} LF(xij,yij)}{Li}}{N} \quad (3)$$

$$FitnessRLWF = \frac{2\times WP\times WR}{WP+WR} \quad (4)$$

where N is the total number of target objects, (xij,yij) is the position of the j-th localization of object i, Li is number of localizations made to object i, WP and WR are the weighted precision and recall, and FitnessRLWF is the localization fitness weighted F-measure, which is used as the fitness function. we chose three image data sets of coins in the experiments. The data sets are intended to provide object localization/detection problems of increasing difficulty. The first data set (*easy*) contains images of coins against an almost uniform background. The second (*medium difficulty*) is of coins against a noisy background, making the task harder. The third data set (*hard*) contains tails and heads of coins against a noisy background.



(a) **(b)** **(c)**

Figure 3: Images in the three data sets. (a) Easy; (b) Medium difficulty; (c) Hard

When a trained localizer is applied to the above images we get localizations as shown in figure 4. For this localization we used window size of 60X60 and step size of 20 which gave us result as per Table 1

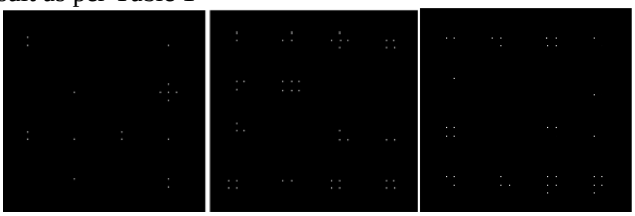


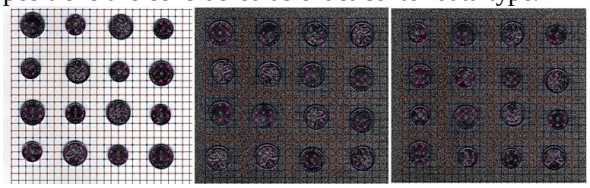
Figure 4: Coins detected locations with window size 60x60 and step size 20 for (a) Coins easy (b) Coins medium (c) Coins hard images

Table1 : Results for fitness function-RLWF with window size 60x60 and step size 20

Image	Weighted recall	Weighted precision	Fitness	Time Elapsed in sec.
coins_easy	0.63	0.45	0.52	4.03
coins_med	0.81	0.41	0.55	2.88
coins_hard	0.88	0.57	0.69	3.90

3.RESULTS FOR OPTIMIZATION OF TRAINING DATA

For optimization of training data, first training data is categorized into four types 1. Exact center 2. Close to center 3. Include center 4. Background data. Figure 5.shows example of how training data is categorized into different data types. When sweeping window moves on image if the center of the window which is tentative coin localization is containing a center of coin then it is considered as close to center, If it contains only small part of object then it is categorized as include center and if no part of the object is within window then it is considered as background type. All the known coin positions are considered as exact center data type.



(a) (b) (c)

Figure 5: Categorization of data into different data Type (a) Easy coin image (b)Medium coin image (c)Hard coin image

For the optimization we used all the exact center data type and percentage from 0 to 100% for all other data types. It was found that exact center type data is important and if the percentage of all other data is used then we get optimal results. For all different percentage of different data types fitness performance for easy, medium and hard coin image was calculated. We calculated relative fitness and time elapsed for training for all three images considering different proportion of four different data types.

In the Figure 6 the x and y axes are the Close to center and Background, and the z axes is the Relative fitness, time elapsed and weighted precision for training for the particular image. Here we observed that for close to center type data the best results can be obtained and when we do not use any example in this type at all then worst results were produced. when percentage of object examples in close to center type is more, the best results are achieved. The Background type objects were not serious for these data sets. Any bad or good influence was not observed by including or excluding these data type. The results suggest that, for these object localization using the relative localization weighted fitness function, good fitness results can be achieved with only the two types of data types, Exact Centre and Close to Centre, and even if not all object examples for the other two types Include Object and Background can be taken out from the training set would not affect much on the result. Also time required for training is less when close to center type data is used then as compared to the other data types as that data type is more as compared to close to center type data.

Table 2 (a) and (b) gives relative fitness and time elapsed using different data proportion for easy coin image and Fig 6 (a) and (b) shows its graphical representation of relative fitness and time elapsed. Here from the table it is observed that when close type percentages is zero (0) and include object type percentage is 100% and background type percentage is ranging from 0 to 100% , less detection is obtained as compared to the detection when maximum percentage of close to center type data is used. From fitness values we observe that as the percentage of close to center data type increases, fitness increases and it does not affect much on the fitness even though we use less percentage of background type and include object type data type. Include object type data and background type data makes very less effect on fitness.

Table 2 (a): Relative fitness for easy coin image.

B\C	0	20	40	60	80	100
0	0.17	0.01	0.01	0.42	0.48	0.64
20	0.19	0.01	0.09	0.40	0.48	0.68
40	0.19	0.20	0.28	0.41	0.48	0.64
60	0.19	0.01	0.28	0.42	0.49	0.64
80	0.13	0.01	0.30	0.42	0.52	0.64
100	0.16	0.06	0.01	0.15	0.50	0.63

Table 2(b): Time elapsed for training for easy coin image

B\C	0	20	40	60	80	100
0	1.99	1.37	1.24	1.44	0.92	0.70
20	1.82	1.39	1.28	1.39	0.93	0.65
40	1.76	1.43	1.28	1.33	0.87	0.66
60	1.96	1.46	1.23	1.44	0.92	0.70
80	1.82	1.40	1.29	1.53	1.15	0.67
100	1.88	1.45	1.24	1.10	0.89	0.68

Table 2(c) : Weighted Precision for easy coin image.

B\C	0	20	40	60	80	100
0	0.08	0.09	0.13	0.18	0.25	0.38
20	0.08	0.08	0.14	0.23	0.30	0.39
40	0.07	0.00	0.12	0.22	0.30	0.38
60	0.08	0.11	0.15	0.23	0.25	0.38
80	0.05	0.00	0.14	0.20	0.30	0.38
100	0.08	0.11	0.15	0.00	0.30	0.38

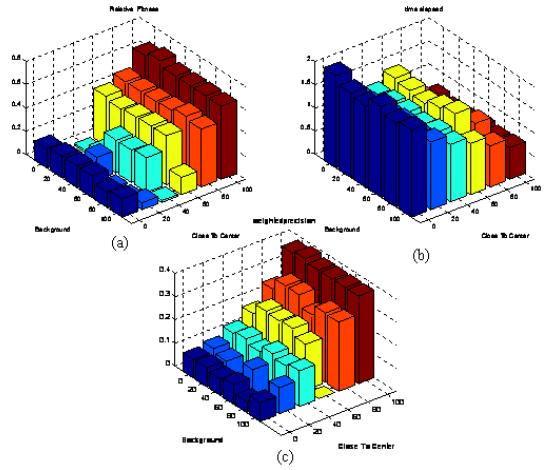


Figure 6.: (a)Relative fitness (b) Time elapsed (c) Weighted precision using different Training Data Proportions for easy coin image

Figure 5 (c). shows the weighted precision for easy coin image with different proportion of training data. It is clearly observed that as percentage of close to center type data is increased weighted precision increases which mean that number of correctly detected objects with respect to number of total detected objects increases. Table 1 (c) gives weighted precision for easy coin image.

Table 3 (a), (b) and (c) gives relative fitness time elapsed and weighted precision using different data proportion for medium coin image and Fig 7 shows its graphical representation. From fitness values it is clear that as the percentage of close to center data type increases detection increases and it does not affect much on the detection even though we use less percentage of background type and include object type data type. Include object type data and background type data makes very less effect on fitness. Also as the percentage of close to center type increases weighted precision increases.

Table 3 (a) : Relative fitness for medium coin image

B\C	0	20	40	60	80	100
0	0.30	0.38	0.41	0.47	0.48	0.68
20	0.33	0.35	0.41	0.47	0.52	0.60
40	0.30	0.38	0.41	0.47	0.48	0.68
60	0.30	0.38	0.41	0.35	0.52	0.60
80	0.17	0.38	0.41	0.47	0.48	0.68
100	0.33	0.38	0.41	0.47	0.48	0.68

Table 3 (b) : Time elapsed for training of medium coin image

B\C	0	20	40	60	80	100
0	2.22	1.81	1.18	1.04	0.81	0.56
20	1.77	1.56	1.68	1.06	0.92	0.59
40	1.78	1.56	1.43	1.20	0.96	0.56
60	1.94	1.38	1.25	1.09	0.78	0.61
80	1.70	1.52	1.21	1.04	0.85	0.56
100	1.90	1.50	1.31	1.16	0.75	0.64

Table 3 (c) : Weighted Precision for medium coin image

B\C	0	20	40	60	80	100
0	0.15	0.21	0.27	0.33	0.38	0.44
20	0.21	0.24	0.27	0.33	0.38	0.45
40	0.15	0.21	0.27	0.33	0.38	0.44
60	0.15	0.21	0.27	0.17	0.38	0.45
80	0.00	0.21	0.27	0.33	0.38	0.43
100	0.21	0.21	0.27	0.33	0.26	0.44

Table 4 (a), (b) and (c) gives relative fitness time elapsed and weighted precision using different data proportion for hard coin image and Fig. 8 shows its graphical representation. From fitness values it is clear that good results are obtained with close to center data type. Include object type data and background type data makes very less effect on fitness. Even though less percentage of background type data and include center type data is used it gives good results. Also as the close to center type data proportion is less so it requires less time for training which reduces training time.

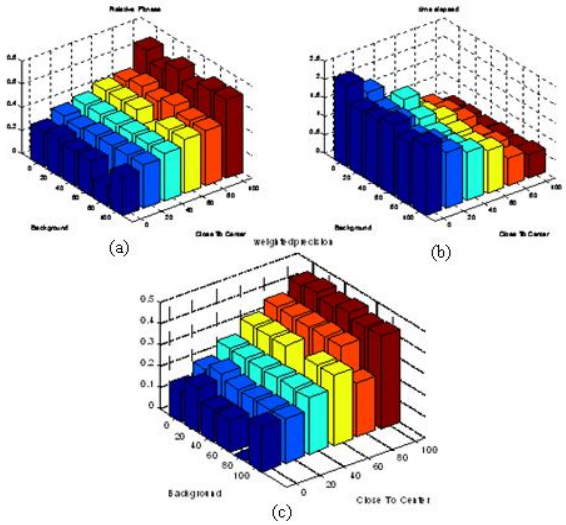


Figure 7: (a) Relative fitness (b) Time elapsed (c) Weighted precision using different Training Data Proportions for medium coin image

Table 4 (a): Relative fitness for hard coin image.

B\C	0	20	40	60	80	100
0	0.25	0.32	0.36	0.40	0.49	0.69
20	0.25	0.32	0.36	0.40	0.49	0.69
40	0.25	0.32	0.36	0.40	0.49	0.69
60	0.25	0.32	0.36	0.40	0.49	0.69
80	0.25	0.32	0.36	0.40	0.49	0.69
100	0.25	0.32	0.36	0.40	0.49	0.56

Table 4(b): Time elapsed for training of hard coin image

B\C	0	20	40	60	80	100
0	1.91	1.62	1.31	1.13	1.06	0.77
20	1.84	1.59	1.28	1.08	1.00	0.79
40	1.96	1.55	1.29	1.07	0.95	0.78
60	2.04	1.62	1.35	1.12	0.90	0.75
80	2.50	1.65	1.32	1.11	0.94	0.61
100	2.02	1.56	1.30	1.12	0.82	0.58

Table 4(c) : Weighted Precision for hard coin image

B/C	0	20	40	60	80	100
0	0.10	0.16	0.22	0.24	0.26	0.45
20	0.10	0.16	0.22	0.24	0.26	0.45
40	0.10	0.00	0.22	0.24	0.26	0.45
60	0.10	0.21	0.22	0.24	0.26	0.45
80	0.10	0.16	0.22	0.24	0.26	0.45
100	0.05	0.16	0.22	0.24	0.26	0.48

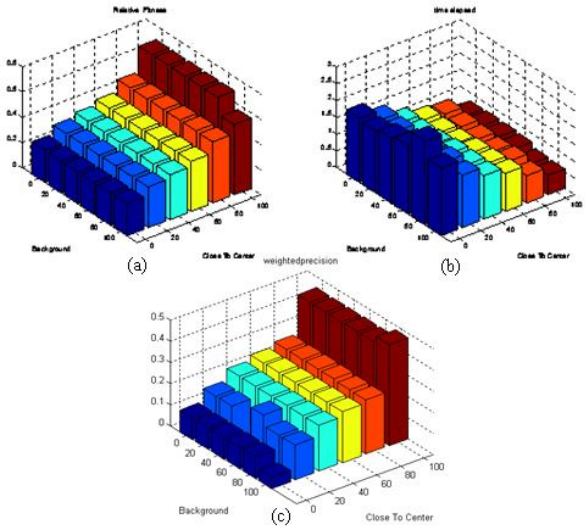


Figure 8: (a)Relative fitness (b) Time elapsed (c) Weighted precision using different Training Data Proportions for hard coin image

4. CONCLUSION

We can categorize a training data into four data types and using different proportion of all these data types to get optimal results which reduces time required for detection of objects. Only Exact center type and close to center type data these two data types can be used to produce good detection results. As background type data contains less information we can neglect that type of data.

Precision & Recall have proved as excellent performance measures and have replaced false alarm and detection rates. Genetic programming has proved that it can be used standalone as an efficient object recognition engine.

REFERENCES

1. J. R. Koza, "*Genetic programming: on the programming of computers by means of natural selection*". London, England: Cambridge, Mass.: MIT Press, 1992.
2. R. Poli, "*Genetic programming for image analysis*," in Proc. 1st Annu. Conf. Genetic Program., J. R. Kosa, D. E. Goldberg, D. B. Fogel, and R. L. Riolo, Eds., Jul. 28–31, 1996, pp. 363–368.
3. Mengjie Zhang, Malcolm Lett. "*Genetic Programming for Object Detection: Improving Fitness Functions and Optimising Training Data"*, IEEE Intelligent Informatics Bulletin (IEEE Computational Intelligence Bulletin). Vol. 7, No. 1. 2006. pp. 12-21.
4. J. F. Winkeler and B. S. Manjunath, "*Genetic programming for object detection*," in Proc. 2ndAnnu. Conf. Genetic Program., J. R. Koza K. Deb, M. Dorigo, D. B. Fogel, M. Garzon, H. Iba, and R. L. Riolo, Eds., Jul. 13–16, 1997, pp. 330–335.
5. M. Zhang, V. Ciesielski, and P. Andreae, "*A domain independent window-approach to multiclass object detection using genetic programming,"*EURASIP Journal on Signal Processing, Special Issue on Genetic and Evolutionary Computation for Signal Processing and Image Analysis, vol. 2003, no. 8, pp. 841–859, 2003. https://doi.org/10.1155/S1110865703303063
6. W. Smart and M. Zhang, "*Classification strategies for image classification in genetic programming,"* in Proceeding of Image and Vision *Computing Conference*, D. Bailey, Ed., Palmerston North, New Zealand, November 2003, pp. 402–407.
7. D. Howard, S. C. Roberts, and R. Brankin, "*Target detection in SAR imagery by genetic programming," Advances in Engineering Software*, vol. 30, pp. 303–311, 1999. IEEE Intelligent Informatics Bulletin December 2006 Vol.7 No.1 https://doi.org/10.1016/S0965-9978(98)00093-3
8. Pritchard, M. Zhang , "*Genetic programming for multi-class object detection"*. Tech. rep. School of Mathematical and Computing Sciences, VUW, 2002.
9. A. Teller and M. Veloso, "*A controlled experiment: Evolution for learning difficult image classification,*" in Proc. 7th Portuguese Conf. Artif. Intell., C. Pinto-Ferreira and N. J. Mamede, Eds., Oct. 3–6, 1995, Lecture Notes in Artificial Intelligence, vol. 990, pp. 165–176. https://doi.org/10.1007/3-540-60428-6_14

10. W. A. Tackett, *"Genetic programming for feature discovery and image discrimination,"* in Proceedings of the 5th International Conference on Genetic Algorithms, ICGA-93, S. Forrest, Ed. University of Illinois at Urbana-Champaign: Morgan Kaufmann, 17-21 July 1993, pp. 303–309.
11. U. Bhowan, *"A domain independent approach to multi-class object detection using genetic programming,"* BSc Honours research thesis, School of Mathematical and Computing Sciences, Victoria University of Wellington, 2003.
12. Bunna NY and M.Zhang *"Multi-class object classification and detection using neural networks"*, Tech. rep.School of Mathematical and Computing Sciences, VUW, 2003.
13. Schneiderman, H., And Kanade, T., *"Object detection using the statistics of parts"*. Int. J. Comput. Vision 56, 3 (2004), 151.177.
14. Barret Chin and Mengjie Zhang, "*Object Detection using Neural Networks and Genetic Programming"* Technical Report CS-TR-07/03. School of Mathematical and Computing Sciences, VUW, November 2007
15. Roy Chow,M.Zhang and PeterAndreae, *" Multiple Class Object Detection Using Pixel Statistics in Neural Networks"*, Technical Report , School of Mathematical and Computing Sciences, VUW, October 2002
16. Will Smart and Mengjie Zhang, "*Applying Online Gradient Descent Search to Genetic Programming for Object Recognition"*, Conferences in Research and Practice in Information Technology,Vol. 32. Australasian Workshop on Data Mining and Web Intelligence(DMWI2004), Dunedin
17. Mengjie Zhang, Will Smart, "*Using Gaussian Distribution to Construct Fitness Functions in Genetic Programming for Multiclass Object Classification"*, Technical Report CS-TR-05/5, School of Mathematical and Computing Sciences, VUW, December 2005.