

Genetic Programming for Multiple Class Object Detection

Mengjie Zhang and Victor Ciesielski

Department of Computer Science
Royal Melbourne Institute of Technology
GPO Box 2476V, Melbourne Victoria 3001, Australia
{mengjie,vc}@cs.rmit.edu.au
<http://www.cs.rmit.edu.au/~{mengjie,vc}>

Abstract. We describe an approach to the use of genetic programming for object detection problems in which the locations of small objects of multiple classes in large pictures must be found. The evolved programs use a feature set computed from a square input field large enough to contain each of objects of interest and applied, in moving window fashion, over the large pictures in order to locate the objects of interest. The fitness function is based on the detection rate and the false alarm rate. We have tested the method on three object detection problems of increasing difficulty with four different classes of interest. On pictures of easy and medium difficulty all objects are detected with no false alarms. On difficult pictures there are still significant numbers of errors, however the results are considerably better than those of a neural network based program for the same problems.

Keywords. Machine learning, Genetic Algorithm, Neural networks, Vision

1 Introduction

As more and more images are captured in electronic form the need for programs which can find objects of interest in a database of images is increasing. For example, it may be necessary to find all tumors in a database of x-ray images, all cyclones in a database of satellite images or a particular face in a database of photographs. The common characteristic of such problems can be phrased as “Given $subpicture_1, subpicture_2 \dots subpicture_n$ which are examples of the object of interest, find all pictures which contain this object and its location(s). Figure 4 show examples of problems of this kind. Figure 4c shows a human retina. We are required to find all of the micro aneurisms and haemorrhages, as indicated by the white squares. Figure 5 shows an enlarged view. Examples of other problems of this kind include target detection problems [5, 17, 19] where the task is to find, say, all tanks, trucks or helicopters in a picture. Unlike most of the current work

¹ In Foo N. (Ed.) *Proceedings of the 12th Australian Joint Conference on Artificial Intelligence* Springer, Heidelberg, P180-191, 1999. ©Springer-Verlag
<http://www.springer.de/comp/lncs/index.html>

in the object recognition area, where the task is to detect only objects of one class [5, 10, 11], our objective is to detect objects from a number of classes.

Several approaches have been applied to automatic object detection and recognition problems. Typically, they use multiple independent stages, such as preprocessing, edge detection, segmentation, feature extraction and object classification [4, 9], which often results in some efficiency and effectiveness problems. The final results rely too much upon the results of each stage. If some objects are lost in one of the early stages stage, it is very difficult or impossible to recover them in the later stage. To avoid these disadvantages, this paper introduces a single stage approach.

There have been a number of reports on the use of genetic programming in object detection [13] and [15]. [18] describes a genetic programming system for object detection in which the evolved functions operate directly on the pixel values. [16] describes a genetic programming system and a face recognition application in which the evolved programs have a local indexed memory. All of these approaches are based on two class problems, that is, objects vs everything else.

Performance in object detection is measured by detection rate and false alarm rate. The detection rate is the number of objects correctly reported as a percentage of the total number of real objects and false alarm rate is the number of objects incorrectly reported as a percentage of the total number of real objects. For example, detection system is looking for grey squares in figure 4a may report that there are 25. If 9 of these are correct the detection rate will be $(9/18) * 100 = 50\%$. The false alarm rate will be $(16/18) * 100 = 88.9\%$ It is important to note that finding objects in pictures with very cluttered backgrounds is an extremely difficult problem and that false detection rates of 200-2,000% (that is the detection system suggests that there are 20 times as many objects as there really are) are common [11, 14].

1.1 Outline of the Approach to Object Detection

A brief outline of the method is as follows:

1. Assemble a database of pictures in which the locations and classes of all of the objects of interest are manually determined. Reserve some of the pictures as ‘unknowns’ for measuring detection performance.
2. Determine an appropriate size (n) of a square which will cover all objects of interest and form the input field.
3. Invoke an evolutionary process to generate a program which can determine the class of an object in its input field.
4. Apply the generated program as a moving window template to the pictures reserved with step 1 and obtain the locations of all the objects of interest in each class. Calculate the the detection rate and the false alarm rate on the test set as the measure of performance.

1.2 Goals

Our overall goal is to determine whether programs evolved by genetic programming can do a good enough job of finding the objects of interest in pictures such as those shown in figure 4. Specifically we are interested in:

- What image processing features would make useful terminals?
- Whether the four standard arithmetic operators will be sufficient for the function set?
- How can the fitness function be constructed given that there are several classes of interest?
- How will performance vary with increasing difficulty of image detection problem?
- Will the performance be better than a neural network approach [21] on the same problems.

2 Genetic Programming

Genetic programming is a relatively recent technology based on the use of Darwinian evolution in the generation of computer programs. Developed and first published by John Koza in [7], it has been successfully applied in areas such as pattern recognition [reference], control, robocup and planning. The process starts with a randomly generated population of programs. Each program is executed and its degree of success in achieving its task is measured and assigned as its fitness. Programs with high fitness are then selected for mating. In the mating process two parents are chosen and randomly selected sub-trees are swapped giving two children of a new population. In general, individuals in the new generation will be fitter than those in the current generation. The process terminates when the best individual does not improve over the course of a few generations.

The programs are constructed from a *terminal set* and a *function set* which will vary according to the problem domain. Functions form the root and the internal nodes of the parse tree representation of a program. Terminals have no arguments and form the leaves of the parse tree. Terminals represent the inputs to the program. Assuming that one has available a generalized genetic programming ‘engine’ to perform the evolutionary processes, the task of using genetic programming for any given problem becomes one of determining the appropriate set of functions and terminals. It is important to note that the selection of the functions and terminals is critical to success. A bad selection could result in very slow convergence or to not being able to find a solution at all. More details of genetic programming and its applications can be found in [2, 7, 8].

2.1 Genetic Programming Adapted to Object Detection

As noted above the main tasks in using genetic programming in some problem domain are (1) Determine the function set, (2) Determine the terminal set, and (3) Determine the method of measuring fitness.

2.2 The Terminal Set

For object detection problems terminals correspond to image features. In our case twenty square feature are extracted from a square input field, which is large enough to contain all objects of interest. In figure 1a, the grey circle is an object of interest, the filled square A1-B1-C1-D1-A1 represents the square input field. Other squares represent various regions of interest. The mean and standard deviation of each local region are used as two separate features. There are 6 local regions giving 12 features. Also we use pixels along the main axes of the input field as shown in figure 1b, giving a total of 20 features.

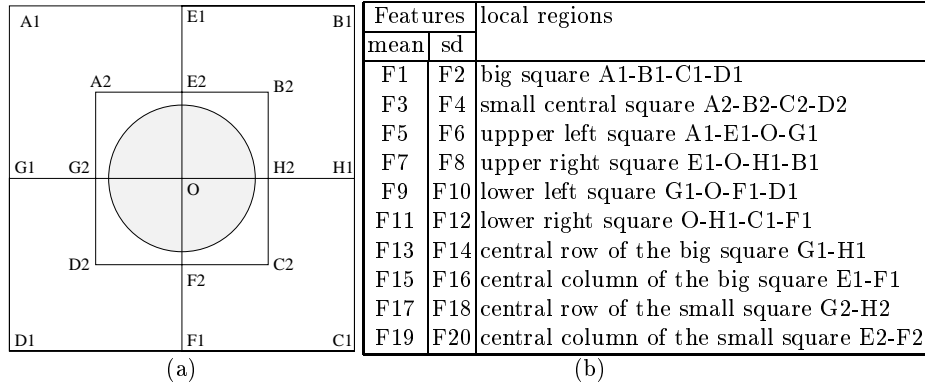


Fig. 1. Feature extraction based on local image region intensities. Note: sd – standard deviation

In addition to these features we have a terminal which generates a random number in the range [0,255]. This corresponds to the number of grey levels in the images; however we have not investigated whether this is significant.

These features have the following characteristics:

- They are symmetrical and contain some information of object translation and rotation invariance.
- Local region features are included. This assists the finding of object centres in the sweeping procedure – if the evolved program is considered as a moving window template the match between the template and the sub image forming the input field will be better when the moving template is close to the centre of an object.
- They are domain independent and easy to extract.

2.3 The Function Set

We use the function set: $F = \{+, -, *, /\}$ which represents four arithmetic operations that form the second order nodes (i.e. 2 arguments)). The +, -, and

* operators have their usual meanings while / represents “protected” division which is the usual division operator except that a divide by zero gives a result of zero. A generated program that performed particularly well is shown in figure 2.

```
(+ (- (+ (+ (/ f16 f14) f5) (+ (/ (/ f11 (* f14 f20)) f11) (- f12
f14))) (- (* (- (* (* (* f9 f11) f1) f10) (* f9 f17)) (/ f5 f18))
(- (+ (+ f17 (* (+ f11 f12) f20)) (* (- (+ f2 145.765) (/ f6 f11))
(- 133.082 f17))) (/ f11 (* f14 f20)))))) (* (- (* (- (- f6 f5) (* f3
f6)) (/ (+ (+ f1 145.765) (* f16 f10)) f18)) f12) (+ (+ f17 (* (+
f17 f12) f20)) (* (+ f14 f12) (- (+ f1 f12) f17))))))
```

Fig. 2. A generated program for the coins problem

The output of any program is a floating point number which must indicate which of the objects of interest is currently in its input field. This is achieved as shown in figure 3 where n is the number of classes of interest, $ProgOut$ is the output value of the evolved program and T is a constant.

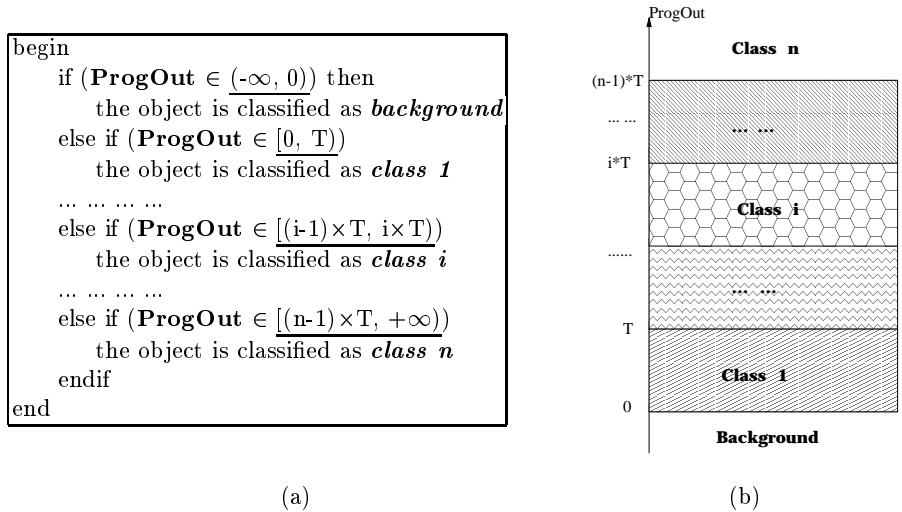


Fig. 3. Mapping of program output to an object classification

2.4 The Fitness Function

The fitness of a program in the population is calculated by using its detection rate and false alarm rate on the training images and is obtained as follows:

1. Apply the program as a moving $n \times n$ (n is the size of the input field) window template to each of the full training images and obtain the output values of the program at each pixel position. Label each pixel position with the ‘detected’ object as described above. Call this data structure a detection map.
2. Find the centres of *objects of interest only*. This is done as follows:
 - Scan the detection map for an object of interest. When one is found mark this point as the centre of the object and continue the scan $n/2$ pixels later.
3. Match these ‘detected’ objects with the known locations of the of each of the ‘true’ objects their classes. A match is considered to occur if the detected object is within TOLERANCE pixels of its known true location. The detection rate Dr and the false alarm rate Fr are then computed.
4. Compute the fitness as shown in equation 1.

$$fitness(Fr, Dr) = A * Fr + B * (1 - Dr) \quad (1)$$

where A and B are constants which reflect the relative importance of false alarm rate versus detection rate.

With this design, it is clear that the smaller the fitness, the better the performance. Zero fitness is the ideal case, which corresponds to the situation in which all of the objects of interest in each class are correctly found by the evolved program without any false alarms.

2.5 Genetic Programming Parameters

The values for the various system parameters used in the experiments are shown in table 1. POPULATION_SIZE is the number of individuals in the population, ELITISM_PCNT gives the percentage of the best individuals in the current population that are copied unchanged to the next generation, CROSS_RATE is the percentage of individuals in the next generation that are to be produced by crossover, MUTATION_RATE is the percentage of individuals in the next generation that are to be produced by mutation (thus ELITISM_PCNT + CROSS_RATE + MUTATION_RATE = 100), CROSS_CHANCE_TERM is the probability that, in a crossover operation two terminals will be swapped, CROSS_CHANCE_FUNC is the probability that in a crossover operation random subtrees will be swapped (thus CROSS_CHANCE_TERM + CROSS_CHANCE_FUNC = 100), INITIAL_MAX_DEPTH is the maximum depth of the randomly generated programs in the initial population, MAX_DEPTH is the maximum depth permitted for programs resulting from crossover and mutation operations, MAX_GENERATIONS gives the stopping condition, T, A, B and TOLERANCE are as described above.

and were taken with a CCD camera over a number of days with relatively similar illumination. In these pictures the background varies slightly in different areas of the image and between images and the objects to be detected are more complex, but still regular. The retina pictures (database 3) were taken by a professional photographer with special apparatus at a clinic and contain very irregular objects on a very cluttered background. The objective is to find two classes of retinal pathologies – haemorrhages and micro aneurisms. blowup picture if room Note that in each of the databases the background counts as a class. The objective is to find the locations (centers) of all the objects of interest in each class.

4 Results

This section presents a series of the experiments on detection problems of increasing difficulty with different classes of interest. The results are compared with those obtained using a neural network approach for object detection on the same databases [20, 21].

4.1 Easy Pictures

Table 2 shows a comparison of the results between the two methods. For Class1 (black circles) and Class3 (grey circles) both methods achieved a 100% detection rate with 0% false alarms. For Class2 (grey squares) the genetic programming system also achieved 100% detection rate with 0% false alarms. However the neural network system had a false alarm rate of 92% with at a detection rate of 100%

Table 2. Comparison of object detection results on 3-class *easy* pictures. Input field size: 14×14

Easy Pictures		Object Classes		
		Class1	Class2	Class3
Best Detection Rate(%)		100	100	100
False Alarm	Neural network	0	92	0
Rate (%)	Genetic programming	0	0	0

4.2 Coin Pictures

Experiments with coin pictures gave similar results which are shown in Table 3. Detecting the heads and tails of 5 cents is relatively straight forward, where the neural network approach led to 100% of detection rate without any false alarms. Detecting heads and tails of 20 cent coins is a difficult problem, where the neural network method resulted in many false alarms. The genetic programming method gave the ideal results, that is, all the objects of interest were found without any false positives for all the four object classes.

Table 3. Comparison of object detection results on 4-class *coin* pictures. Input field size: 24×24

Coin Pictures		Object Classes			
		head005	tail005	head020	tail020
Best Detection Rate(%)		100	100	100	100
False Alarm	Neural network	0	0	182	37.5
Rate (%)	Genetic programming	0	0	0	0

4.3 Retina Pictures

The results for the retina pictures are summarised in Table 4. Compared with the results for the previous image databases the results are disappointing. However, the false alarm rate is greatly improved over the neural network method.

The results over the three data bases show similar trends: the genetic programming method always gives a lower false alarm rate for the same detection rate.

Table 4. Comparison of object detection results on retina pictures. Input field size: 16×16

Retina Pictures		Object Classes	
		haem	micro
Best Detection Rate(%)		70	100
False Alarm	Neural network	2698	10104
Rate (%)	Genetic programming	1357	588

4.4 Summary and Analysis

Summary We have tested the GP based approach on multi-class object detection problems of increasing difficulty: a three-class easy detection problem, a four-class medium difficulty coin detection problem and a very difficult two-class retinal pathology detection problem. In all cases on easy and medium difficulty detection problems the new approach produced the ideal results, that is, all the objects of interest in every class were found without any false alarms. For “micro” and “haem” detection in the very difficult retina pictures, the new method resulted in much better performance than the neural network approach.

Analysis of Results of Retina Picture We found three main reasons why the results on retina pictures are not as good as those on easy and coin pictures. Firstly, in easy and coin pictures the background is relatively uniform, whereas in the retina pictures it is highly cluttered. Secondly, in the retina pictures, there are only two classes of interest, that is, “micro” and “haem”, but there are also several other classes such as veins and other eye anatomy. Thus the objects of non-interest are classified background. It appears that this makes the

background too complex to be considered as a single class. Thirdly, in the easy and coin pictures all of the objects in a class are the same size whereas the sizes of the objects in each class in the retina pictures are quite different. The size of “micro” varies from 3×3 to 5×5 pixels and the sizes of “haem” varies from 7×7 to 16×16 pixels. Thus the sizes of “micro” are still similar, but this is not the case for the “haem”. This might be the main reason that the results for detecting “micro” are much better than that of “haem”. This suggests that the current set of functions and terminals cannot be successfully applied to detecting objects in the same class with a large variation in size.

Further Experiments We experimented with an alternative set of terminals based on a circular set of features. The features were computed based on a series of concentric circles centred in input field. This terminal set focused on boundaries rather than regions. In the coin and retina pictures, the results based on this terminal set are slightly worse than those with the square region feature set. This suggests that the local region features are better for these detection problems.

We also experimented with a different function set. We hypothesised that convergence might be quicker if the function values were close to the range $(-1,1)$. We used *dabs*, *sin* and *exp*, (absolute value, sine and exponent to base e) in addition to the four arithmetic operators (*+*, *-*, *** and */*). The detection results are similar to those based on the function set of only the four arithmetic operators. Convergence was slightly faster training the coin and retina pictures, but slightly slower for easy pictures. This suggests that *dabs*, *sin* and *exp* may be useful for more difficult problems, however considerably more experimentation is required.

5 Conclusions

The goal of this paper was to develop a general, single stage method for detecting small objects of multiple classes in large pictures based on genetic programming. A secondary goal was to compare the performance of this method with a neural network based method. The results show that genetic programming can be used to generate programs for object detection. The method appears to be applicable to detection problems of varying difficulty. The genetic programming method also gives better detection performance than a neural network based approach for the same problem set.

The genetic programming approach has the following limitations:

- The training times for the coin problem and the retina problem are quite long. Some of the runs took longer than 48 hours on a 4 processor ULTRA-SPARC4. We are investigating ways of shortening the training times.
- The method is not particularly effective in detecting objects with different sizes which are in the same class, for example the haemorrhages in the retina pictures. This might be related to the simple pixel-based image features used.

High level image features which contain size invariant information need to be investigated.

- The classification strategy employed in the generated programs is not easy to determine. However, if a particular feature does not appear in a program that works well, it can be inferred that that feature is not important.
- Some experimentation is required to find good values of the various parameters for each different problem.

Acknowledgements

We would like to thank Dr. James Thom and Dr. Zhi-Qiang Liu for useful discussions on image processing and retrieval, Peter Wilson whose basic genetic programming package was used in the project and Chris Kamusinski who provided and labelled the retina pictures.

References

1. Ballard, D.H., Brown, C.M.: Computer Vision. Englewood Cliffs, N.J: Prentice-Hall, Inc.(1982)
2. Banzhaf, W., Nordin, P., Keller, R.E. and Francone, F.D.: Genetic Programming: An Introduction on the Automatic Evolution of Computer Programs and Its Applications. Morgan Kaufmann, Inc. San Francisco, California(1998)
3. Bernardon, A., and Carrick, J.E.: A neural system for automatic target learning and recognition applied to bare and camouflaged sar targets. Neural Networks, Vol.8. **7/8** (1995) 1103–1108
4. Casasent, D.P., Neiberg, L.M.: Classifier and Shift-invariant Automatic Target Recognition Neural Networks. Neural Networks, Vol.8. **7/8** (1995) 1117–1129
5. Gader, P.D., Miramonti, J.R., Won Y., Coffield P.: Segmentation free shared weight neural networks for automatic vehicle detection. Neural Networks, Vol.8. **9** (1995)1457–1473
6. Hinton, G.E.: Connectionist learning procedures. Artificial Intelligence, Vol. 40. **1** (1989)185–234
7. Koza, J.R.: Genetic programming : on the programming of computers by means of natural selection. Cambridge, Mass. : MIT Press, London, England(1992)
8. Koza, J.R.: *Genetic programming II: Automatic Discovery of Reusable Programs*. Cambridge, Mass.: MIT Press, London, England(1994)
9. Rogers, S.K., Colombi, J.M., Martin, C.E., Gainey, J.C., Fielding, K.H., Burns, T.J., Ruck, D.W., Kabrisky, M., Ocley, M.: Neural networks for automatic target recognition. Neural Networks, Vol.8. **7/8** (1995)1153–1184
10. Roitblat, H.L., Au, W.W.L., Nachtigall, P.E., Shizumura,R., Moons, G.: Sonar Recognition of Targets Embedded in Sediment. Neural Networks, Vol. 8. **7-8** (1995) 1263–1273
11. Roth, M.W.,: Survey of neural network technology for automatic target recognition. IEEE Transactions on neural networks, Vol. 1. **1** (1990) 28–43
12. Rumelhart,D.E., Hinton,G.E.,Williams,R.J.: Learning internal representations by error propagation. In: Rumelhart,D.E., McClelland,J.L. (eds): Parallel distributed Processing, Explorations in the Microstructure of Cognition, Volume 1: Foundations. Chap.8. The MIT Press, Cambridge, Massachusetts, London, England(1986)

13. Sherrah, J.R., Bogner, R.E., Bouzerdoum, A.: The evolutionary pre-processor: Automatic feature extraction for supervised classification using genetic programming. In: Koza, J.R., Deb, K., Dorigo, M., Fogel, D.B., Garzon, M., Iba, H., Riolo, R.L. (eds.): Genetic Programming 1997: Proceedings of the Second Annual Conference, Stanford University, Morgan Kaufmann (1997) 304–312
14. Shirvaikar, M., Trivedi, M.: A network Filter to detect small targets in high Clutter backgrounds. *IEEE Transactions on Neural Networks*, Vol. 6. **1** (1995) 252–257
15. Tackett, W.A.: Genetic Programming for Feature Discovery and Image Discrimination. In: Forrest S. (ed.): Proceedings of the 5th International Conference on Genetic Algorithms (ICGA-93), Morgan Kaufmann, University of Illinois at Urbana-Champaign (1993) 303–309
16. Teller A., Veloso, M.: A controlled experiment : Evolution for learning difficult image classification. In: Pinto-Ferreira, C., Mamede, N.J. (eds): Proceedings of the 7th Portuguese Conference on Artificial Intelligence, Berlin, Springer Verlag, LNAI, vol.990. (1995)165–176
17. Waxman, A.M., Seibert, M.C., Gove, A., Fay, D.A., Bernandon, A.M., Lazott, C., Steele, W.R., Cunningham, R.K.: Neural Processing of Targets in Visible, Multispectral IR and SAR Imagery. *Neural Networks*, Vol. 8. **7-8** (1995)1029-1051
18. Winkeler, J.F., Manjunath, B.S.: Genetic Programming for Object Detection. In: Koza, J.R., Deb, K., Dorigo, M., Fogel, D.B., Garzon, M., Iba, H., Riolo, R.L. (eds.): Genetic Programming 1997: Proceedings of the Second Annual Conference, Morgan Kaufmann, Stanford University, (1997) 330–335
19. Won, Y., Gader, P.D., Coffield, P.C.: Morphological shared-weight networks with applications to automatic target recognition. *IEEE Transactions on Neural Networks*, Vol.8., **5**(1997)1195–1203
20. Zhang, M.: A pixel-based approach to recognising small objects in large pictures using neural networks. In: Proceedings of the Annual RMIT Computer Science Post-graduate Students' Conference, Department of computer science, RMIT. TR 97-51, Melbourne (1997) 57-68
21. Zhang, M., Ciesielski, V.: Centred Weight Initialization in Neural Networks for Object Detection. In: Proceedings of the Twenty Second Australasian Computer Science Conference, Auckland (1999)

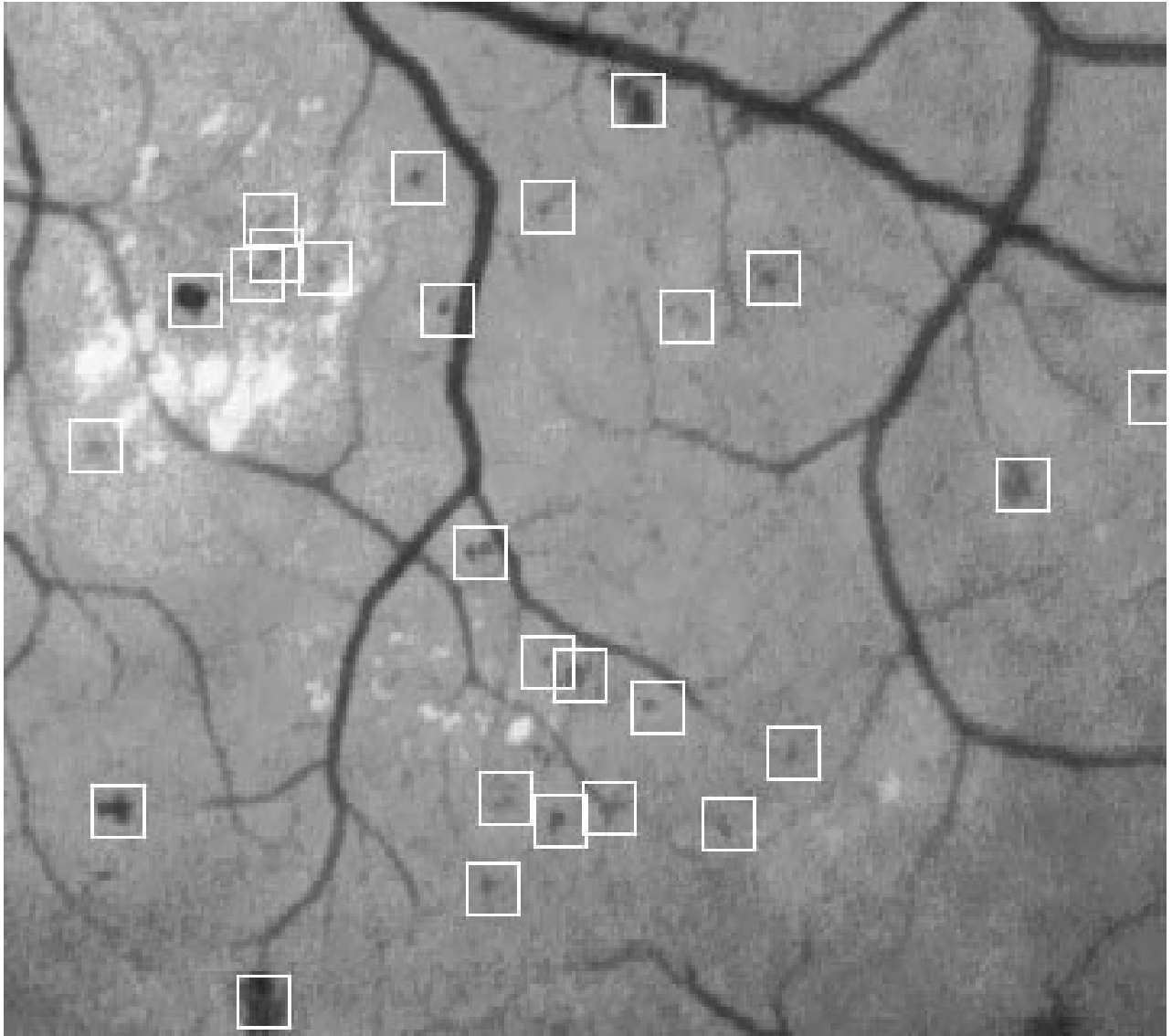


Fig. 5. Enlargement of retina picture