Evolving Novel Image Features Using Genetic Programming-based Image Transforms

Taras Kowaliw, Wolfgang Banzhaf, Nawwaf Kharma, and Simon Harding

***Abstract*— In this paper, we use Genetic Programming (GP) to define a set of transforms on the space of greyscale images. The motivation is to allow an evolutionary algorithm means of transforming a set of image patterns into a more classifiable form. To this end, we introduce the notion of a Transform-based Evolvable Feature (TEF), a moment value extracted from a GP-transformed image, used in a classification task. Unlike many previous approaches, the TEF allows the whole image space to be searched and augmented. TEFs are instantiated through Cartesian Genetic Programming, and applied to a medical image classification task, that of detecting muscular dystrophy-indicating inclusions in cell images. It is shown that the inclusion of a single TEF allows for significantly superior classification relative to predefined features alone.**

I. Introduction

In this paper, we explore means of augmenting the typical use of features in classification problems. Due to increased capacity for computation, we may now, in a training phase, explore means of searching image databases directly for novel database-specific features. Hence, rather than applying only a general set of predefined features, or devoting significant human effort to the careful design of database-specific features, we may use techniques of machine learning in the creation of new features well suited to a given classification problem.

Here, we will attempt to evolve transformations on the space of images, in hopes that particular transforms which emphasize distinguishing characteristics may be found. These transforms will be genetic programs (GPs), optimized through an evolutionary algorithm. A set of moments describing the transformed image will be extracted, and used for classification. The introduction of GP-based transforms to the classification task brings two desireable properties to the pattern recognition process: firstly, it allows for the search of the entire pixel-space of images, not just a collection of predefined features, potentially finding patterns missed by human designers; and secondly, it allows for recombination of this pixel-space view of the database into forms useful to a classifier. This is reminiscent of aspects of Support Vector Machines, where additional dimensions are added to a problem space through a kernel to improve classifiability: here, we do so on the pixel space using a GP kernel.

Our domain of application will be a binary classification problem based on the detection of a form of Muscular Dystrophy in cell nuclei. A database of pre-segmented cell images has been procured, and we will attempt to train a classifier to recognize human-defined ground truth. We will show that the addition of a single evolved TEF to our database will improve recognition by 38% relative to a set of predefined features often used for cell classification, obviously allowing evolution to successfully exploit our additional information sources. In so doing, we will provide evidence for the efficacy of an evolutionary design of image features, and the potential for the automation of pattern recognition.

II. Review

A. Predefined Features and Feature Selection for Image Classification

Image classification (recognition) typically uses a set of features to reduce the dimensionality of the pattern space. A great deal of effort is spent on the design, selection and weighting of features. Often, practitioners begin with a set of "standard" features, then use some machine learning technique to find the most appropriate choices; these "standard" features often involve statistical moments, or entropy- or histogram-based measures (as in [1]). Domain specific measures are used as well, such as the nucleus-specific measures in the construction of the Wisconsin Breast Cancer Database [2].

Let an image space (some finite, non-trivial rectangle of pixel locations) be denoted I, and an image on that space be denoted $f(I)$. Each pixel $p = (p_x, p_y) \in I$ is associated with an intensity, $f(p) \in [0, 1]$. Moments are statistical descriptions of some given data, often used to break down an image into numeric values which can be sent to a machine learning technique. In discrete form, *geometric moments* are often computed as

$$M_{mn} = \sum_{p \in I} p_x^m p_y^n f(p)$$

Central moments are computed similarly, although relative to the centre of mass:

$$C_{mn} = \sum_{p \in I} (p_x - \bar{x})^m (p_y - \bar{y})^n f(p)$$

where $(\bar{x}, \bar{y}) = (M_{10}, M_{01})$ is the centroid. From these moments it is possible to define a set of scale- and rotation-invariant moments, known as Hu's moments [3].

The *entropy* of an image is typically defined to be the entropy of the image histogram. That is, the histogram $(\bar{h})$ is

Taras Kowaliw, Wolfgang Banzhaf, and Simon Harding are with the Department of Computer Science, Memorial University of Newfoundland, St. John's, NL, Canada, A1B 3X5. Nawwaf Kharma is with the Department of Electrical and Computer Engineering, Concordia University, 1455 de Maisonneuve Blvd. Ouest, Montréal, QC, Canada, H3G 1M8. Correspondence to taras@kowaliw.ca

interpreted as a probability distribution on the space of pixel values, then the Shannon entropy is computed:

$$E(\bar{h}) = \sum_{i=0}^{255} -h_i \log h_i$$

where we have assumed an 8-bit greyscale image. Entropy can be normalized by dividing by the maximum value, here $\log 256$.

The *fractal dimension* of an image, as computed through a box-counting algorithm, is a rough measure of self-similarity of an image. First, the image is thresholded, to produce a black and white version. Next, a sequence of boxes are computed, those boxes being used as discretizations of the image. For each box size, the number of boxes required to cover the image is computed. Finally, the log over the box sizes is plotted against the log of the number of boxes: the coefficient of the line of best fit (by linear regression) is considered the fractal dimension.

Following previous work with cell classification [2], we also may define several features specifically for cell images. Given a pre-segmented cell image, we may define the *perimeter*, *compactness*, *mean radius*, *radius variance*, and *compactness* of the cell boundary. Further, having divided the image into nuclear interior and exterior, we can compute the *area* and greyscale *variance* of the interior.

B. Feature Creation

By feature creation, we refer to the creation of numerical methods drawn from the raw images in a database. This is a contrast with some usage, which refers to the pre-processing of values of predefined features prior to classification (as in [4]).

The most significant research dealing with the issue of feature creation may be divided into four categories.

Historically speaking, the first category contains methods that did not use artificial evolution in any form. This category include the work of [5], [6], [7], [8]. All of them used mask- or pixel-based features; evaluated them using a reference pattern; had no or limited higher-level feature generation; and were not ready for real-world application.

The second category comprises methods that employed polynomial functions of features extracted from the target pattern. These include [9], [10], [11]. All of these techniques employ primitive features that are statistical functions of discrete (or discretizable) data signals; create complex features, in the form of polynomial functions of the primitive features; and most have applications in the area of machine fault diagnosis.

A third category covers those methods that involved true Genetic Programming techniques. The most notable early efforts in this area include those of [12], [13]. Both methods evolve a LISP-style program which controls a roving agent that uses any or all of five evolvable Boolean functions or masks to correctly identify a character. What distinguishes these works is that they evolve both the features and the classifier that uses them.

The final and most modern category is that of Evolvable Pattern Recognizers. These are perhaps the most ambitious research projects. Two efforts are worthy of special mention: HELPR [14] and CellNet [1], [15]. HELPR evolves feature detectors, but the classification module is completely separate from the feature extraction module. CellNet blurs the line between feature selection and the construction of binary classifiers out of these features. Other differences exist but both attempts are the only systems, that we know of, that aim at using artificial evolution to synthesize complete recognition systems with minimum human intervention.

C. Genetic Programming and Image Processing

Cartesian Genetic Programming was originally developed by Miller and Thomson [16] for the purpose of evolving digital circuits. It represents a program as a directed graph, or, in our specific case, provides a graph mapping inputs to an output. One of the benefits of this type of representation is the implicit re-use of nodes in the directed graph. The technique is also similar to Parallel Distributed GP, which was independently developed by Poli [17], and also to Linear GP developed by Banzhaf *et al* [18]. Harding and Banzhaf have used Cartesian Genetic Programming to reverse-engineer common image filters [19], [20]. CGP has also been used to configure the logic blocks in specialized, parallel hardware developed, applied to the re-evolution of known image filters [21], [22]. Evolved image filters have been used in real-world problems, such as the detection of mud slides [23].

III. The Model

Here we describe the use of CGP as a means of defining transformations on the space of images, and the use of these transformations as evolvable features. Theoretically, nearly any form of Genetic Programming could have been used, CGP was chosen due to previous image processing success. Further, although we apply to medical images, we expect this approach to be applicable to a wider range of image classification tasks.

We use CGP as described by Harding and Banzhaf [19], utilizing a one-dimensional topology, and infinite levels-back. Each node in the graph has two connections to previous nodes or inputs (x and y), and uses the function set $\{max\{x,y\}$, $min\{x,y\}$, $x \cdot y$, $x+y$, x/y, $x-y$, $const.$, $|x|$, x^2, $x^y\}$, where division is "safe" (i.e. returns 1 when the denominator is very close to 0). The number of inputs is a system parameter, and drawn from a square neighbourhood surrounding a given pixel. The inputs are drawn from this square neighbourhood from the top left corner, left to right, top to bottom. The output is the value of the final node in the graph.

An example image of a CGP with 16 inputs is shown in Figure 1. Note several forms of "neutral" code: the green (second), blue (third), and orange (fourth) nodes use one or less of their two inputs; the grey (second-last) node does not connect to the pink (final) output node. Removing

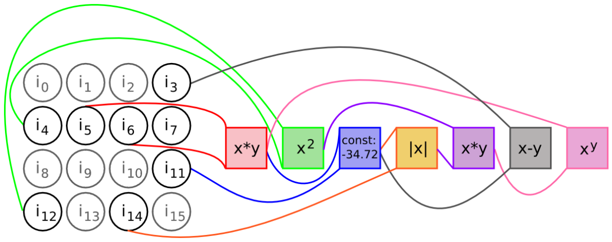


Fig. 1. Illustration of a CGP graph with 16 inputs (circles) and 7 nodes (squares). The output is the value in the final node.

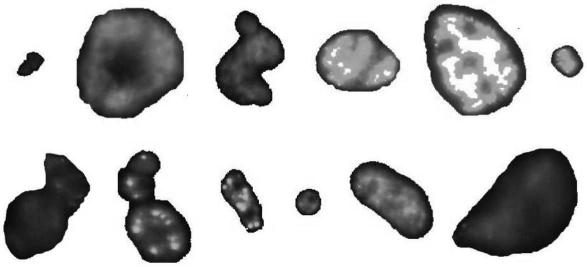


Fig. 2. Examples of healthy *(top row)* and sick *(bottom row)* cell images.

redundancy, this CGP reduces to the function:

$$\phi(i_5, i_6, i_{12}) = (i_5 i_6)^{(34.72(i_{12})^2)}$$

CGP graphs are easily evolvable. Crossover is defined classically (i.e. single-point swap), treating the graph as a list of nodes. Mutation operates on the node types, connectivity, and neighbourhood size, treating each element in turn with equal probability.

A. CGP Image Transformations

We consider the application of a CGP graph to a pixel and its neighbourhood. As a neighbourhood, we will define squares of integer size surrounding the sources pixel.

We shall write the output of an arbitrary CGP graph G applied to a list of k values as $G(x_1, ..., x_k)$. Given some location $p \in I$, we apply CGP graph G to pixel p and neighbourhood of size n, denoted $\bar{p} = \{p_0, p_1, ..., p_{n^2}\}$, as follows:

$$G(f(\bar{p})) = G(f(p_0), f(p_1), ..., f(p_{n^2})) \qquad (1)$$

i.e. we retrieve the pixel values $f(p_i)$ for $i = 1, ..., n^2$ in the square neighbourhood surrounding pixel p, and feed them to the CGP graph in order. Note that if $f(q)$ does not exist for some pixel q (beyond the edges of the image, say), then we return the value $f(q) = -1$. Further, if $G(f(\bar{p})) \notin [0, 1]$, we replace by the closest boundary value, either 0 or 1.

Given some image $f(I)$, we can define a new image, $G(f(I))$ as follows: Let I' be an image space of same dimensions as I. For each pixel $p' \in I'$, let $f(p') = G(f(\bar{p}))$. Hence, every CGP graph G can be viewed as a function on the space of images.

This is but one of many different sorts of filter functions, and hence is a limit on the space of what can be (easily) represented. Generalization is a future endeavour of ours, but beyond the scope of this research.

IV. Image Patterns and Evaluation

Features are evaluated on the basis of their ability to separate images in a target database. The database chosen, CellsDB[1], was collected by the Centre hospitalier de l'Université de Montréal (CHUM), where the causes and associated symptoms for Oculopharyngeal Muscular Dystrophy (OPMD) at the genetic and cellular level have been studied extensively [24]. Intranuclear inclusions (INIs) have been detected via both pathological studies and electron microscopy; These INIs were tubular, about 8.5 nm in external and 3 nm in inner diameters, up to 0.25 μm in length, and converged to form tangles or palisades [25]. Detection of these inclusions is expected to lead to the detection of OPMD. Hence, we seek to be able to find features which separate images of cells on the basis of whether or not they contain INIs. Detecting INIs, as opposed to other intranuclear patterns, is a difficult task, requiring training for human classification.

CellsDB was collected and prepared by Tarundeep Dhot at the Centre for the Study of Brain Diseases at CHUM, and is pre-segmented so that each image contains a single cell; It is a collection of images of cells taken at 10x, 20x, and 40x zoom, divided into two categories associated with the presence or absence of inclusions indicating OPMD: "healthy" and "sick". An example of some cell images may be seen in Figure 2. Data from the CellsDB has previously been used for image processing [26], [27].

We wish to award fitness to any particular feature based on its ability to distinguish between healthy and sick cells. To do so, we convert the database to a set of features, then attempt to classify the cells using some given classifier and evaluation technique. The classifier will return a false positive rate, FPR, for both classes, which we combine to a measure of *sensitivity-specificity*.

$$SS = (1 - FPR(\text{“healthy”}))(1 - FPR(\text{“sick”})) \qquad (2)$$

Hence, $SS \in [0, 1]$ is maximized for perfect recognition. It is trivial to include weights for the various classes into such a function, as may be necessary for medical application, and it is unlikely that evolvability will be affected by such weighting.

The features and evaluation method changed between training and validation runs. The pattern database was broken into two sets: 186 healthy and 200 sick cell images for training, and 200 healthy and 200 sick cell images for validation. All classifiers and evaluation techniques were implemented via the Weka machine learning system, version 3.5.7 [28].

[1] Since undertaking this work, it was shown that some of the images in CellsDB were misclassified: In proceeding discussions, five images from the training set, and two images from the validation set were mislabeled. This will not affect reported validation accuracies by more than 1%.

For training runs, a set of 16 features were computed for the TEF-transformed images: $\{M_{00}, M_{10}, M_{01}, C_{11}, C_{20}, C_{02}, C_{12}, C_{21}, C_{22}, H_1, ..., H_7\}$, where H_i is Hu's ith invariant moment. These 16 features of transformed images were trained and evaluated on the training set of images using 5-fold cross validation.

For validation runs, a set of 40 features were computed: Firstly, the $16 + 8$ predefined moments applied to untransformed images, consisting of the 16 aforementioned moments, along with the set $\{threshholded\ area, variance, mean\ radius, radius\ variance, perimeter, compactness, entropy, fractal\ dimension\}$. Secondly, the 16 aforementioned moments applied to the transformed images. These 40 features were trained and evaluated on the validation set of images using 10-fold cross validation.

The reason that a smaller set of features was used in the definition of the training evaluation is as follows: randomization of the database prior to classification is a useful feature to help prevent overfitting, but leads to a stochastic fitness function. Beginning with a set of features which is already adept at classification, as the predefined features are, makes the variance due to stochasticity greater than fitness gains in early evolution, hence hindering the selection operator.

V. Experiments

A. Initial Classification Experiments

Initial classification attempts were undertaken using the 24 predefined features and moments on CellsDB. The classifiers used included Decision Trees, Ridor rules, and k-NN. These were run 40 times and averaged to include the variance associated with randomization of the database patterns. Next, we evaluated the performance of randomly-generated TEFs under classification. For each of the above classifiers, a randomly generated TEF was instantiated, and the 16 additional moments generated by the transform were added to the database. This was repeated 40 times, and averaged. The results are shown in Table I.

As can be seen, there is little difference between the SS value for simply the pre-defined features and moments, and the inclusion of a randomly defined TEF. However, the variance of values increases under the addition of a TEF, as the new feature values allow for helpful or misleading / confusing information to the classifier. Note the additional time requirements for additional moment calculations, which clearly dominate the process.

Several feature evaluation routines were used in an attempt to improve classification accuracy (Information gain attribute evaluation, χ^2 attribute evaluation, and principal component evaluation), but none significantly improved performance (indeed, most significantly lessened classifier accuracy).

B. TEF Evolution

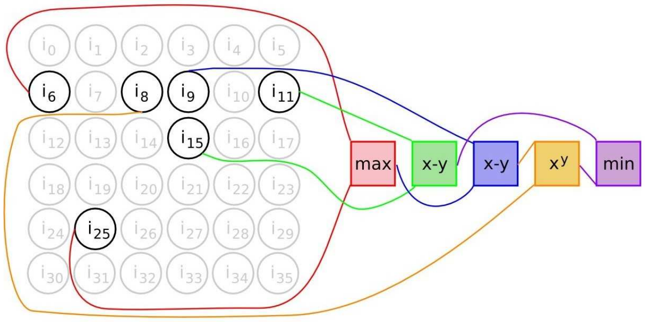
We used a standard evolutionary algorithm, as described by Eiben and Smith [29]. Following an informal parameter search, the following parameters were chosen for the evolutionary algorithm:

initial pop. size	400	*pop. size*	200
prob. mutation	0.02	*prob. crossover*	0.6
prop. elite	0.01	*tournament size*	3
mask size	6×6	*CGP graph size*	100

Each EA was run for 50 generations, using a 1-NN classifier. 40 runs were undertaken in total.

Evolution was quite successful at increasing fitness (training SS), which increased from a mean best fitness of 0.523 in the first generation (averaged over 40 runs, s.d. 0.026), to a mean best fitness of 0.620 (s.d. 0.046). In a few cases, evolution optimized training SS at the expense of validation SS, but a good general increase in the latter was seen in most runs. Mean best validation SS for the final generation was 0.676 (s.d. 0.052), and maximized at a value of 0.766. This is a 32% improvement over the expected performance of the best classifier found for predefined features alone.

C. Best Discovered Transform



i_0 i_1 i_2 i_3 i_4 i_5
i_6 i_7 i_8 i_9 i_{10} i_{11}
i_{12} i_{13} i_{14} i_{15} i_{16} i_{17}
i_{18} i_{19} i_{20} i_{21} i_{22} i_{23}
i_{24} i_{25} i_{26} i_{27} i_{28} i_{29}
i_{30} i_{31} i_{32} i_{33} i_{34} i_{35}
max
x-y
x-y
x^y
min

Fig. 3. Illustration of best discovered CGP.

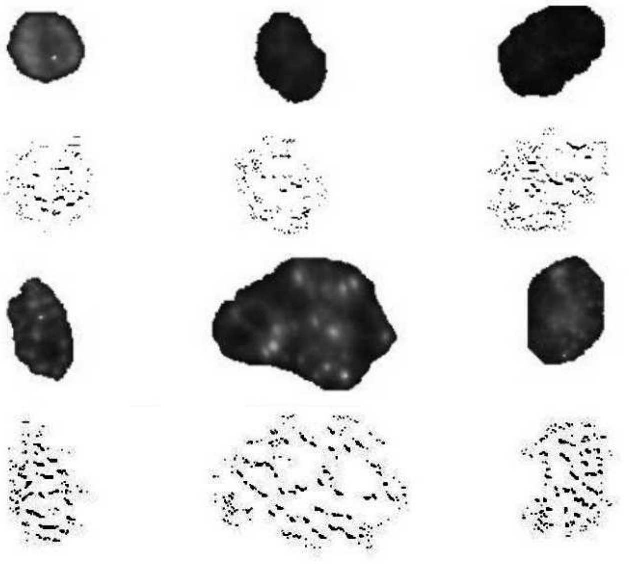


Fig. 4. Examples of output of best discovered TEF: View of original cell images *(rows one and three)* and images subjected to action by CGP *(rows two and four)*. Top rows are healthy, bottom rows are sick. *(Transformed images have been subjected to contrast stretching and colour inversion to make distinction between black and dark grey more visible.)*

TABLE I

COMPARISON OF CLASSIFICATION ON STANDARD DATABASE OF UNEVOLVED FEATURES ($\emptyset$) TO DATABASE AUGMENTED BY A RANDOMLY-GENERATED TEF. ALL FIGURES AVERAGED OVER 40 RUNS, WITH STANDARD DEVIATION IN BRACKETS.

classifier	**transform**	SS	$TPR(H)$	$TPR(S)$	**time *ms***
decision tree	$\emptyset$	0.4945 (0.0201)	0.7223 (0.0349)	0.6858 (0.0335)	369.3 (101.1)
	random TEF	0.5213 (0.0800)	0.7236 (0.0633)	0.7177 (0.0534)	77883.4 (9135.9)
ridor	$\emptyset$	0.4606 (0.0236)	0.7025 (0.0841)	0.6646 (0.0829)	473.3 (109.6)
	random TEF	0.5034 (0.0586)	0.7061 (0.0735)	0.7172 (0.0825)	76374.3 (12943.5)
1-NN	$\emptyset$	0.5166 (0.0117)	0.7323 (0.0137)	0.7056 (0.0126)	601.9 (73.2)
	random TEF	0.5527 (0.0698)	0.7585 (0.0472)	0.7262 (0.0464)	99860.2 (16031.5)
5-NN	$\emptyset$	0.5803 (0.0157)	0.7735 (0.0182)	0.7503 (0.0122)	174.4 (99.6)
	random TEF	0.6108 (0.0670)	0.7993 (0.0582)	0.7623 (0.0302)	77368.5 (13933.3)

TABLE II

COMPARISON OF CLASSIFICATION RESULTS FOR UNEVOLVED VERSUS BEST EVOLVED FEATURES.

classifier	**transform**	SS	$TPR(H)$	$TPR(S)$
5-NN	$\emptyset$	0.580	0.774	0.750
1-NN	best TEF	0.766	0.878	0.873
decision tree	best TEF	0.801	0.912	0.878

In this section, we have extracted the best evolved transform and will explore it in more detail. The best individual of the final (50th) generation of the highest validation fitness run was selected. This best individual had a sensitivity-specificity of 0.766, encompassing true positive rates of 0.878 for healthy cells and 0.873 for sick cells.

The same evolved attribute values, when evaluated using a J48 Decision Tree instead of 1-NN, gives a sensitivity-specificity of 0.801, encompassing true positive rates of 0.912 for healthy cells and 0.878 for sick cells, or a 38% improvement over the expected performance of the best classifier found for predefined features alone. Performance for the best discovered TEF, relative to the best expected performance for the unevolved features alone, is summarized in Table II

The best transform, once neutral code is removed, may be written as the following neighbourhood function:

$$o(i_0, ..., i_{35}) = \min\{i_{11} - i_{15}, (i_9 - \max\{i_6, i_{25}\})^{i_8}\}$$

The graph view is shown in Figure 3. This function returns black (0) nearly always, except when both $i_{11} - i_{15}$ and $(i_9 - \max\{i_6, i_{25}\})^{i_8}$ are (relatively) high. The first subsection ensures variance on the right of the neighbourhood is high (excluding inclusions that are too large), and the second subsection ensures that the left and bottom-left are mostly white, excluding inclusions too small. Hence, we detect the right half of inclusions of the proper size and variance. Note that this function works for several different microscope zoom levels simultaneously.

While non-OPMD indicating inclusions are still highlighted using this function, the OPMD-indicating inclusions are highlighted with more intensity, hence allowing for a greater recognition of the cells than original image features alone. Examples of output are shown in Figure 4.

Feature selection was run on the database of the original moments and the evolved moments, using an Information Gain Attribute Evaluator. The evaluator selected 30 of the features as significant, discarding the other 10. The top ten ranked attributes, by information gain, were all moments of the evolved transformed image.

VI. CONCLUSIONS AND FUTURE DIRECTIONS

In this paper, we have defined a mechanism for the automated discovery of novel database-specific features via evolutionary computation. This system has been applied to a new and significant recognition problem in medical imaging, and shown to significantly improve overall performance. In the short term, we have discovered a set of features that will achieve a 91% recognition for healthy cell images, and a 88% recognition for sick cell images, a 38% improvement over predefined features alone. More generally, we have achieved a significant advance in the capacity of automated systems to automatically adapt to a given pattern database without human expertise, or, perhaps, with greater efficacy than human designs.

To properly evaluate the power of this new approach, several future directions immediately suggest themselves: (a) to apply the same framework to several different databases, showing both generality and the capacity for automated adaptation; and (b) to extend the system to use a collection of features rather than a single graph, or to evolve the choice of moments simultaneously. These steps bear potential to advance the goal of a fully-automated pattern recognition system. Both steps are currently being explored at the Memorial University of Newfoundland.

Acknowledgements

The authors would like to greatfully acknowledge the work of the Centre hospitalier de l'Université de Montréal and Mr. Tarundeep Dhot for provision of the image database. Travel support has been provided by the National Science and Engineering Research Council of Canada under contract DG 283304-07.

References

[1] N. Kharma, T. Kowaliw, E. Clement, C. Jensen, A. Youssef, and J. Yao, "Project cellnet: Evolving an autonomous pattern recognizer," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 18, no. 6, pp. 1039–1056, 2004.

[2] W. Street, W. Wolberg, and O. Mangasarian, "Nuclear feature extraction for breast tumor diagnosis," in *IS&T/SPIE 1993 International Symposium on Electronic Imaging*, 1993.

[3] M. K. Hu, "Visual pattern recognition by moment invariants," *IRE Transactions on Information Theory*, vol. IT-8, pp. 179–187, 1962.

[4] M. Smith and L. Bull, "Using genetic programming for feature creation with a genetic algorithm feature selector," in *Parallel Problem Solving from Nature - PPSN VIII*, 2004.

[5] P. Gader and M. Khabou, "Automatic feature generation for handwritten digit recognition," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 18, no. 12, 1996.

[6] A. M. Gillies, "Automatic generation of morphological template features," in *Proceedings of SPIE - The International Society for Optical Engineering*, vol. 1350, 1990, pp. 252–261.

[7] F. W. M. Stentiford, "Automatic feature design for optical character recognition using an evolutionary search procedure," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-7, no. 3, pp. 350–355, 1985.

[8] L. Uhr and C. Vossler, "A pattern -recognition program that generates, evaluates and adjusts its own operators," in *Computers and Thought*. McGraw-Hill, 1963.

[9] E. I. Chang, R. P. Lippmann, and D. W. Tong, "Using genetic algorithms to select and create features pattern classification," in *IJCNN International Joint Conference*, vol. 3, 1990, pp. 747–753.

[10] P. Chen, T. Toyota, and Z. He, "Automated function generation of symptom parameters and application to fault diagnosis of machinery under variable operating conditions," *IEEE Transactions on Systems, Man and Cybernetics-Part A: Systems and Humans*, vol. 31, no. 6, pp. 775–781, 2001.

[11] H. Guo, L. Jack, and A. Nandi, "Feature generation using genetic programming with application to fault classification," *IEEE Transactions on Systems, Man and Cybernetics — Part B: Cybernetics*, vol. 35, no. 1, 2005.

[12] D. Andre, "Automatically defined features: The simultaneous evolution of 2-dimensional feature detectors and an algorithm for using them," in *Advances in Genetic Programming*, K. E. Kinnear, Ed. MIT Pres, 1994.

[13] J. Koza, "Simultaneous discovery of detectors and a way of using the detectors via genetic programming," in *IEEE International Conference on Neural Networks*, vol. 3, 1993, pp. 1794–1801.

[14] M. M. Rizki, M. A. Zmuda, and L. A. Tamburino, "Evolving pattern recognition systems," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 6, pp. 594–609, 2002.

[15] T. Kowaliw, N. Kharma, C. Jensen, H. Moghnieh, and J. Yao, "Using competetive co-evolution to evolve better pattern recognizers," *International Journal of Computational Intelligence and Applications*, vol. 5-3, pp. 305–320, 2005.

[16] J. F. Miller and P. Thomson, "Cartesian genetic programming," in *Proc. EuroGP 2000*, ser. LNCS, R. Poli, W. Banzhaf, W. B. Langdon, J. F. Miller, P. Nordin, and T. C. Fogarty, Eds., vol. 1802. Springer-Verlag, 2000, pp. 121–132.

[17] R. Poli, "Parallel distributed genetic programming," in *New Ideas in Optimization*, D. Corne, M. Dorigo, and F. Glover, Eds. McGraw-Hill, 1999.

[18] W. Banzhaf, P. Nordin, R. Keller, and F. Francone, *Genetic Programming - An Introduction. On the Automatic Evolution of Computer Programs and its Application*. Morgan Kaufmann, 1998.

[19] S. Harding, "Evolution of image filters on graphic processor units using cartesian genetic programming," in *WCCI 2008*, 2008.

[20] S. Harding and W. Banzhaf, "Genetic programming on GPUs for image processing," in *Proc. First International Workshop on Parallel and Bioinspired Algorithms (WPABA-2008)*, F. F. J. Lanchares and J. Risco-Martin, Eds., 2008, pp. 65–73.

[21] Z. Vacek and L. Sekanina, "Evaluation of a new platform for image filter evolution," in *Proc. of the 2007 NASA/ESA Conference on Adaptive Hardware and Systems*, 2007.

[22] P. N. Kumar, S. Suresh, and J. R. P. Perinbam, "Digital image filter design using evolvable hardware," in *ICIS 05: Proceedings of the Fourth Annual ACIS International Conference on Computer and Information Science*, 2004.

[23] P. Rosin and J. Hervas, "Image thresholding for landslide detection by genetic programming," in *Analysis of multi-temporal remote sensing images,*, L. Bruzzone and P. Smits, Eds. World Scientific, 2002, pp. 65–72.

[24] B. Brais, J. P. Bouchard, Y. G. Xie, D. L. Rochefort, N. Chretien, F. M. Tome, R. G. Lafreniere, J. M. Rommens, E. Uyama, and O. Nohira, "Short GCG expansions in the PABP2 gene cause oculopharyngeal muscular dystrophy," *Nature Genetics*, vol. 18, pp. 164–167, 1998.

[25] F. M. S. Tome and M. Fradeau, "Nuclear changes in muscle disorders," *Methods Achiev Exp Pathol*, vol. 12, pp. 261–296, 1986.

[26] J. Yao, N. Kharma, and P. Grogono, "A multi-population genetic algorithm for robust and fast ellipse detection," *Pattern Analysis and Application*, vol. 8, pp. 149–162, 2005.

[27] N. N. Kharma, H. Moghnieh, J. Yao, Y. Guo, A. Abu-Baker, J. Laganiere, G. Rouleau, and M. Cheriet, "Automatic segmentation of cells from microscopic imagery using ellipse detection," *IET Image Processing*, vol. 1, no. 1, pp. 39–47, 2007.

[28] H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques, 2nd Ed.* Morgan Kaufman, 2005.

[29] A. E. Eiben and J. E. Smith, *Introduction to Evolutionary Computing*. Springer, 2003.