PalmPrints: A Novel Co-Evolutionary Algorithm For Clustering Finger Images

Nawwaf Kharma[1], Ching Y. Suen, and Pei F. Guo

Departments of Electrical & Computer Engineering and Computer Science,

Concordia University, 1455 de Maisonneuve Blvd. West, Montreal, QC, H3G 1M8, Canada

[1] kharma@ece.concordia.ca

Abstract. The purpose of this study is to explore an alternative means of hand image classification, one that requires minimal human intervention. The main tool for accomplishing this is a Genetic Algorithm (GA). This study is more than just another GA application; it introduces (a) a novel cooperative co-evolutionary clustering algorithm with dynamic clustering and feature selection; (b) an extended fitness function, which is particularly suited to an integrated dynamic clustering space. Despite its complexity, the results of this study are clear: the GA evolved an average clustering of 4 clusters, with minimal overlap between them.

1 Introduction

Biometric approaches to identity verification offer a mostly *convenient* and potentially *effective* means of personal identification. All such techniques, whether palm-based or not, rely on the individual's most-unique and stable, physical or behavioural characteristics.

The use of multiple sets of features requires feature selection as a prerequisite for the subsequent application of classification or clustering [5, 8]. In [5], a hybrid genetic algorithm (GA) for feature selection resulted in (a) better convergence properties; (b) significant improvement in terms of final performance; and (c) the acquisition of subset-size feature control. Again, in [8], a GA, in combination with a k-nearest neighbour classifier, was successfully employed in feature dimensionality reduction.

Clustering is the grouping of similar objects (e.g. hand images) together in one set. It is an important unsupervised classification technique. The simplest and most well known clustering algorithm is the k-means algorithm. However, this algorithm requires that the user specifies, before hand, the desired number of clusters. An evolutionary strategy implementing variable length clustering in the x-y plane was developed to address the problem of dynamic clustering [3]. Additionally, a genetic clustering algorithm was used to determine the best number of clusters, while simultaneously clustering objects [9].

Genetic algorithms are randomized search and optimization techniques guided by the principles of evolution and natural genetics, and offering a large amount of implicit parallelism. GAs perform search in complex, large and multi-modal landscapes. They have been used to provide (near-)optimal solutions to many optimization problems [4].

Cooperative co-evolution refers to the simultaneous evolution of two or more species with coupled fitness. Such evolution allows the discovery of complex solutions wherever complex solutions are needed. The fitness of an individual depends on its ability to collaborate with individuals from other species. In this way, the evolutionary pressure stemming from the difficulty of the problem favours the development of cooperative individual strategies [7].

In this paper, we propose a cooperative co-evolutionary clustering algorithm, which integrates dynamic clustering, with (hand-based) feature selection. The co-evolutionary part is defined as the problem of partitioning a set of hand objects into a number of clusters without *a priori* knowledge of the feature space. The paper is organized as follows. In section 2, hand feature extraction is described. In section 3, cooperative co-evolutionary clustering and feature selection are presented, along with implementation results. Finally, the conclusions are presented in section 4.

2 Feature Extraction

Hand geometry refers to the geometric structure of the hand. Shape analysis requires the extraction of object features, often normalized, and invariant to various geometric transformations such as translation, rotation and (to a lesser degree) scaling. The features used may be divided into two sets: geometric features and statistical features.

2.1 Geometric Features

The geometrical features measured can be divided into six categories:

- Finger Width(s): the distance between the minima of the two phalanges at either side of a finger. The line connecting those two phalanges is termed the finger base-line.
- Finger Height(s): the length of the line starting at the fingertip and intersecting (at right angles) with the finger base-line.
- Finger Circumference(s): The length of the finger contour.
- Finger Angle(s): The two acute angles made between the finger base-line and the two lines connecting the phalange minima with the finger tip.
- Finger Base Length(s): The length of the finger base-lines.
- Palm Aspect Ratio: the ratio of the ‘palm width’ to the ‘palm height’. Palm width is (double) the distance between the phalange joint of the middle finger, and the midpoint of the line connecting the outer points of the base lines of the thumb and pinkie (call it *mp*). Palm length is (double) the shortest distance between *mp* and the right edge of the palm image.

2.2 Statistical Features

Before any statistical features are measured, the fingers are re-oriented (see Fig. 1), such that they are standing upright by using the Rotation and Shifting of the Coordinate Systems. Then, each 2D finger contour is mapped onto a 1D contour (see Fig. 2), taking the finger midpoint centre as its reference point. The shape analysis for four fingers (excluding the thumb) is measured using: (1) Central moments; (2) Fourier descriptors; (3) Zernike moments.

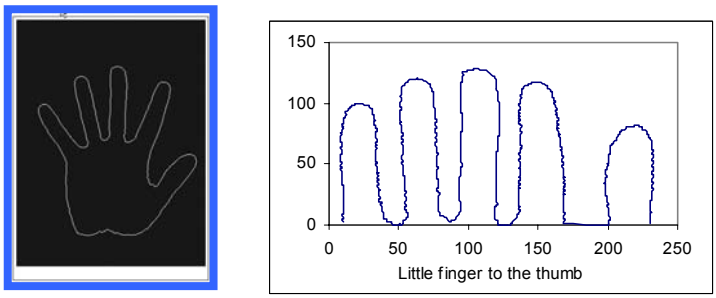

Fig. 1. Hand Fingers (vertically re-oriented) using the Rotation and Shifting of the Coordinate Systems

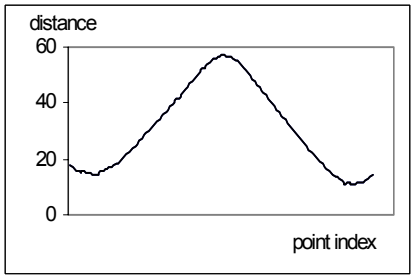

Fig. 2. 1D Contour of a Finger. The y-axis represents the Euclidean distance between the contour point and the finger midpoint centre (called the reference point)

Central Moments. For a digital image, the p_{th} order regular moment with respect to a one-dimensional function *F[n]* is defined as:

$$R_p = \sum_{n=0}^{N} n^p \cdot F[n]$$

The normalized one-dimensional p_{th} order central moments are defined as:

$$M_p = \sum_{n=0}^{N} (n - \bar{n})^p \cdot F[n] \qquad \bar{n} = R_1 / R_0$$

F[n]: with n∈[0,N]; the Euclidean distance between point n and the finger reference point.
N: the total number of pixels.

Fourier Descriptors. We define a normalized cumulative function Φ* as an expanding Fourier series to obtain descriptive coefficients (Fourier Descriptors or FD's). Given a periodic 1D digital function *F[n]* in [0, N] points (periodic), the expanding Fourier series is:

$$\Phi^*(t) = \frac{a_0}{2} + \sum_{k=1}^{\infty} (a_k \cos\frac{2\pi k}{N} \cdot t + b_k sin\frac{2\pi k}{N} \cdot t)$$

$$a_k = \frac{2}{N}\sum_{n=1}^{N} F[n] \cdot \cos\frac{2\pi k}{N} \cdot n \ , \qquad b_k = \frac{2}{N}\sum_{n=1}^{N} F[n] \cdot sin\frac{2\pi k}{N} \cdot n$$

The k_{th} harmonic amplitudes of the Fourier Descriptors are:

$$A_k = \sqrt{a_k^2 + b_k^2} \qquad k = 1,2, ..$$

Zernike Moments. For a digital image with a polar form function $f(\rho,\varphi)$, the normalized $(n+m)_{th}$ order Zernike moments is approximated by:

$$Z_{nm} \approx \frac{n+1}{N}\sum_j f(\rho_j,\varphi_j) \cdot V_{nm}^*(\rho_j,\varphi_j)$$

$$V_{nm}(\rho,\varphi) = R_{nm}(\rho) \cdot e^{jm\varphi} \ , \quad x_j^2 + y_j^2 \le 1$$

$$R_{nm}(\rho) = \sum_{s=0}^{(n-|m|/2)} \frac{(-1)^s (n-s)!\rho^{n-2s}}{s!((n+|m|)/2-s)!((n-|m|)/2-s)!}$$

n: a positive integer.
m: a positive or negative integer subject to the constraints that n-|m| is even, |l|≤ n.
$f(\rho_j,\varphi_j)$: the length of vector between point j and the finger reference point.

3 Co-evolution in Dynamic Clustering and Feature Selection

Our clustering application involves the optimization of three quantities, which together form a complete solution, (1) the set of features (dimensions) used for clustering; (2) the actual cluster centres; and (3) the total number of clusters. Since this is the case, and since the relationship between the three quantities is complementary (as opposed to adversarial), it makes sense to use cooperative (as

opposed to competitive) co-evolution as the model for the overall genetic optimization process. Indeed, it is our hypothesis that whenever a (complete) potential solution (i) is comprised of a number of complementary components; (ii) has a medium-high degree of dimensionality; and (iii) features a relatively low level of coupling between the various components; then attempting a cooperative co-evolutionary approach is justified.

In similarity-based clustering techniques, a number of cluster centres are proposed. An input pattern (point) is assigned to the cluster whose centre is closest to the point. After all the points are assigned to clusters, the cluster centres are re-computed. Then, the points are re-assigned to the (new) clusters based (again) on their distance from the new cluster centres. This process is iterative, and hence it continues until the locations of the cluster centres stabilize.

During co-evolutionary clustering, the above occurs, but in addition, less discriminatory features are eliminated, leaving a more efficient subset for use. As a result, the overall output of the genetic optimization process is a number of traditionally good (i.e. tight and well-separated) clusters, which also exist in the smallest possible feature space.

The co-evolutionary genetic algorithm used entails that we have two populations (one of cluster centres and another of dimension selections: more on this below), each going through a typical GA process. This process is iterative and follows these steps:
(a) fitness evaluation; (b) selection; (c) the application of crossover and mutation (to generate the next population); (d) convergence testing (to decide whether to exit or not); (e) back to (a).

This continues until the convergence test is satisfied and the process is stopped. The GA process is applied to the first population and in parallel (but totally independently) to the second population. The only difference between a GA applied to one (evolving population) and a GA applied to two cooperatively co-evolving populations is that fitness evaluation of an individual in one population is done after that individual is joined to another individual in the other population. Hence, the fitness of individuals in one population is actually coupled with (and is evaluated with the help of) individuals in the other population.

Below, is a description of the most important aspects of the genetic algorithm applied to the co-evolving populations that make-up PalmPrints. First, the way individuals are represented (as chromosomes) is described. This is followed by an explanation of step (a) to step (e), listed above. Finally, a discussion of the results is presented.

3.1 Chromosomal Representation

In any co-evolutionary genetic algorithm, two (or more) populations co-evolve. In our case, there are only two populations, (a) a population of cluster centres (*Cpop*), each represented by a variable-length vector of real numbers; and (b) a population of 'dimension-selections', or simply dimensions (*Dpop*), each represented by a vector of bits. Each individual in *Cpop* represents a (whole) number of cluster centre coordinates. The total number of coordinates equals the number of clusters. On the other hand, each individual ('dimension-selection') in *Dpop* indicates, via its '1' bits,

which dimensions will be used and which, via its '0' bits, will not be used. Splicing an individual (or chromosome) from *Cpop* with an individual (or chromosome) from *Dpop* will give us an overall chromosome that has the following form:

$$\{(A1, B1, \dots, Z1), (A2, B2, \dots, Z2), \dots (An, Bn, \dots, Zn), 10110\dots0\}$$

Taken as a single representational unit, this chromosome determines:

(1) The *number of clusters*, via the number of cluster centres in the left-hand side of the chromosome;
(2) The actual *cluster centres*, via the coordinates of cluster centres, also presented in the left-hand side of the chromosome; and
(3) The *number of dimensions* (or features) used to represent the cluster centres, via the bit vector on the right-hand side of the chromosome.

As an example, the chromosome presented above has n clusters in three dimensions: the first, third and fourth dimensions. (This is so because the bit vector has 1 in its first bit location, 1 in its third bit location and 1 in its fourth bit location.) The maximum number of feature dimensions (allowed in this example) is equal to the number of letters in the English alphabet: 26, while the minimum is 1. And, the maximum number of clusters (which is not shown) is $m>n$.

3.2 Crossover and Mutation, Generally

In our approach, the crossover operators need to (a) deal with varying-length chromosomes; (b) allow for a varying number of feature dimensions; (c) allow for a varying number of clusters; and (d) to be able to adjust the values of the coordinates of the various cluster centres. This is not a trivial task, and is achieved via a host of crossover operators, each tuned for its own task. This is explained below.

Crossover and Mutation for Cpop. *Cpop* needs crossover and mutation operators suited for variable-length clusters as well as real-valued parameters**.** When crossing over two parent chromosomes to produce two new child chromosomes, the algorithm follows a three-step procedure:

(a) The length of a child chromosome is randomly selected from the range: [*2*, *MaxLength*], where *MaxLength* is equal to the total number of clusters in both parent chromosomes;
(b) Each child chromosome picks up copies of cluster centre coordinates, from each of the two parents, in proportion to the relative fitness of the parents (to each other); and finally,
(c) The actual values of the cluster coordinates are modified using the following (mutation) formula for *i*th feature with α randomly selected from the range [0,1]:

$$f_i = \min(Fi) + \alpha\,[\max(Fi) - \min(Fi)]\,. \qquad (1)$$

Fi: the ith feature dimension, i= 0,1,2….
α: a random value ranged [0,1].
*min(f*i*) / max(f*i*): minimum / maximum value that feature i can take.*

With α changed within [0,1], the function of equation (1) varies the ith feature dimension in its own feature distinguished range [*min(Fi)*, *max(Fi)*] as for the variation of actual values of the cluster coordinates (see Fig. 3).

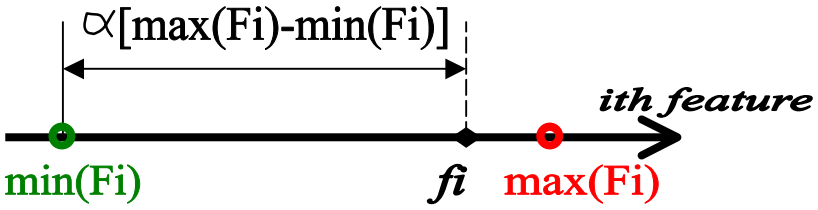

Fig. 3. Variation of the *ith* feature dimension within [*min(Fi)*, *max(Fi)*] with a random value α ranged [0,1]

In addition to crossover, mutation is applied, with a probability μ_c to one set of cluster centre coordinates. The value of μ_c used is 0.2 (or 20%).

Crossover and Mutation for Dpop. *Dpop* needs one crossover operator suited for fixed length binary-valued parameters. For a binary representation of *Dpop* chromosomes, single-point crossover is applied. Following that, mutation is applied with a mutation rate of μ_d. The value of μ_d used is 0.02.

3.2 Selection and Generation of Future Generations

For both populations, elitism is applied first, and causes copies of the fittest chromosomes to be carried over (without change) from the current generation to the next generation. Elitism is set at 12% of *Cpop* and 10% of *Dpop*. Another 12% of *Cpop* and 10% of *Dpop* are generated via the crossing over of pairs of elite individuals, to generate an equal number of children. The rest (76% of *Cpop* and 80% of *Dpop*) of the next generation is generated through the application of crossover and mutation (in that order) to randomly selected individuals from the non-elite part of the current generation. Crossover is applied with a probability of 1 (i.e. all selected individuals are crossed over), while mutation is applied with a probability of 20% for *Cpop* and 2% for *Dpop*.

3.3 Fitness Function

Since the Mean Square Error (MSE) can always be decreased by adding a data point as a cluster centre, fitness was a monotonically decreasing function of cluster numbers. The fitness function (MSE) was poorly suited for comparing clustering situations that had a different numbers of clusters. A heuristic MSE was chosen with dynamic cluster *n*, based on the one given by [3].

In our own approach of dynamic clustering with feature selection in a co-evolutionary GA, there are two dynamic variables interchanged with the two populations: dynamic clustering and dynamic feature dimensions. Hence, a new *extended* MSE fitness is proposed for our model, which measures quantities of both object tightness (f_T) and cluster separation (f_S):

$$MSE \quad extended \quad fitness = \sqrt{n+1}(f_T + \frac{1}{f_S})$$

$$f_T = \sum_{i=1}^{n}\sum_{j=1}^{mi} d(c_i, x_j^i)/n, \quad f_S = \sqrt{k+1}\sum_{i=1}^{n} d\{c_i, Ave(\sum_{j=1, j\neq i}^{n} c_j)\}$$

n: *dynamic no. of clusters*
k: *dynamic no. of features*
c_i: *the ith cluster centre*
Ave(A): *the average value of A*
mi: *the number of data points belonging to the* i_{th} *cluster*
x_j^i: *the* j_{th} *data point belonging to the* i_{th} *cluster*
d(a,b): *the Euclidean distance between points a and b*

The square root of the number of clusters and the square root of the number of dimension in *MSE extended fitness* are chosen to be unbiased in the dynamic co-evolutionary environment. The point of the MSE extended fitness is to optimize of the distance criterion by minimizing the within-cluster spread and maximizing the inter-cluster separation.

3.4 Convergence Testing

The number of generations prior to termination depends on whether an acceptable solution is reached or a set number of iterations are exceeded. Most genetic algorithms keep track of the population statistics in the form of population maximum and mean fitness, standard deviation of (maximum or mean) fitness, and minimum cost. Any of these or any combination of these can serve as a convergence test. In PalmPrints, we stop the GA when the maximum fitness does not change by more than .001 for 10 consecutive generations.

3.5 Implementation Results

The *Dpop* population is initialized with 500 members, from which 50 parents were paired from top to bottom. The remaining 400 offspring are produced randomly using

single-point crossover and a mutation rate (μ_d) of 0.02. *Cpop* is initialized at 88 individuals, from which 10 members are selected to produce 10 direct new copies in the next generation. The remaining 68 are generated randomly, using the dimension fine-tuning crossover strategy and a mutation rate (μ_c) of 0.2.

The experiment presented here uses 100 hand images and 84 normalized features. Termination occurred at a maximum of 250 generations, since it is discovered that fitness converged to less than 0.0001 variance prior. The results are promising; the average co-evolutionary clustering fitness is 0.9912 with a significantly low standard deviation of 0.1108. The average number of clusters is 4, with a very low standard deviation of 0.4714. Average hand image misplacement rate is 0.0580, with a low standard deviation of 2.044. Following convergence, the dimension of the feature space is 41, with zero standard deviation. Hence, half of the original 84 features are eliminated. Convergence results are shown in Fig. 4.

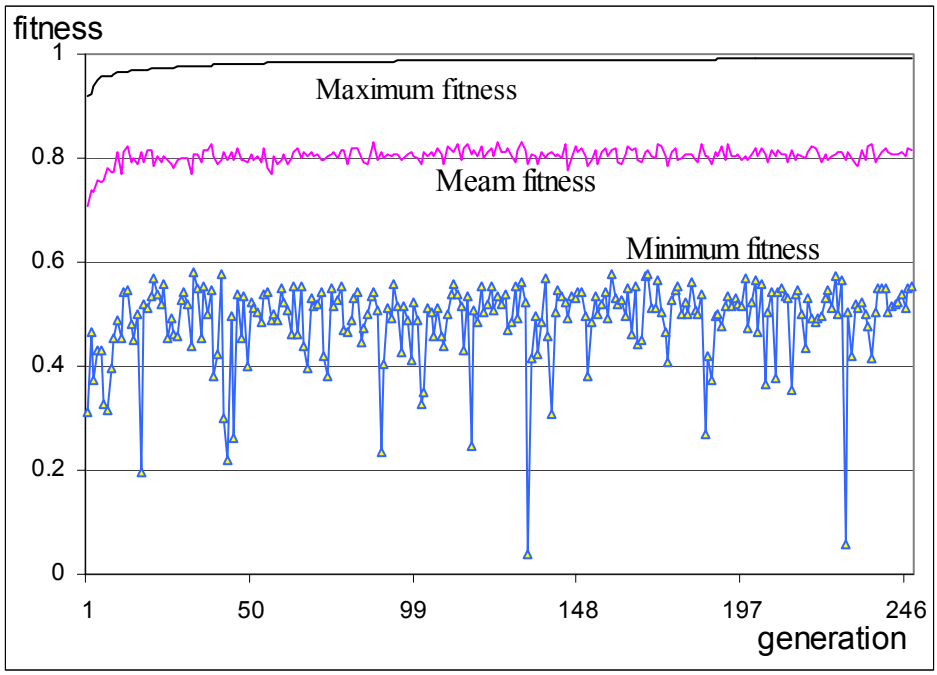

Fig. 4. Convergence results

4 Conclusions

This study is the first to use a genetic algorithm to simultaneously achieve dimensionality reduction and object (hand image) clustering. In order to do this, a cooperative co-evolutionary GA is crafted, one that uses two populations of part-

solutions in order to evolve complete highly fit solutions for the whole problem. It does succeed in both its objectives. The results show that the dimensionality of the clustering space is cut in half. The number (4) and quality (0.058) of clusters produced are also very good. These results open the way towards other cooperative co-evolutionary applications, in which 3 or *more* populations are used to co-evolve solutions and designs consisting of 3 or more loosely-coupled sub-solutions or modules.

In addition to the main contribution of this study, the authors introduce a number of new or modified structural (e.g. palm aspect ratio) and statistical features (e.g. finger 1D contour transformation) that may prove equally useful to others working on the development of biometric-based technologies.

References

1. Fogel, D.B.: Evolutionary Computation: Toward A New Philosophy Of Machine Intelligence. IEEE Press, New York, (1995) ***5***
2. Haupt, R.L. and Haupt, S.E.: Practical Genetic Algorithms. Wiley Interscience, New York (1998)
3. Lee, C.-Y.: Efficient Automatic Engineering Design Synthesis Via Evolutionary Exploration. PhD thesis (2002), California Institute of Technology, Pasadena, California
4. Maulik, U., Bandyopadhyay S.: Genetic Algorithm-based Clustering Technique. Pattern Recognition 33 (2000) 1455-1465
5. Oh, I.-S., Lee, J.-S. and Moon, B.-R.: Local Search-embedded Genetic Algorithms For Feature Selection. Proc. of International Conf. on Pattern Recognition (2002) 148-151
6. Paredis, J.: Coevolutionary Computation. Artificial Life 2 (1995) 355-375
7. Pena-Reyes, C.A., Sipper M.: Fuzzy CoCo: A Cooperative-Coevolutionary Approach To Fuzzy Modeling. IEEE Transaction on Fuzzy Systems Vol. 9, No.5 (October 2001) 727-737
8. Raymer, M.L., Punch, W.F., Goodman, E.D., Kuhn, L.A. and Jain, A.K.: Dimensionality Reduction Using Genetic Algorithms. IEEE Transaction on Evolutionary Computation, Vol. 4, No.2 (July 2000) 164 -171
9. Tseng, L.Y., Yang, S.B.: A Genetic Approach To The Automatic Clustering Problem. Pattern Recognition 34 (2001) 415-424