Intelligent Zoning Design Using Multi-Objective Evolutionary Algorithms

†Paulo V. W. Radtke[1,2], Luiz S. Oliveira[1], Robert Sabourin[1,2], Tony Wong[1]
[1]École de Technologie Supérieure - Montreal, Canada
[2]Pontifícia Universidade Católica do Paraná - Curitiba, Brazil
†e-mail: radtke@livia.etsmtl.ca

Abstract

This paper discusses the use of multi objective evolutionary algorithms applied to the engineering of zoning for handwriten recognition. Usually a task fulfilled by an human expert, zoning design relies on specific domain knowledge and a trial and error process to select an adequate design. Our proposed approach to automatically define the zone design was tested and was able to define zoning strategies that performed better than our former strategy defined manually.

1 Introduction

The first step for handwriting recognition is to determine an appropriate representation for the handwriten symbols [8], a process usually made by an human expert and refined on a trial and error basis. Zoning has been used for this task often, as in [4], allowing the analysis of local information based on the partitioning of the symbol image. This paper presents an automatic approach to define the zoning for offline handwriten recognition, using Multi-Objective Evolutionary Algorithms [2].

Multi-objective evolutionary algorithms – MOEAs – have been proven useful in many applications in different domains. These algorithms are based on a Darwinian search process, where a population of candidate solutions evolve through generations by means of genetic operators, such as selection, cross-over and mutation. While researching for this application we found other two related works, [9] and [6], using Genetic Programming, a different aproach to evolutionary algorithms.

The first work is also on offline handwriten recognition, but the approach employs active pattern recognition instead of zoning and can not be compared directly. The second, while being an online approach, uses zones to extract features, and allow us for some comparisons. There are some differences on the approach when compared to ours, as using a non regular zoning strategy and allowing overlapping, or genetic programming to search the solutions. While the two approaches are not directly comparable, we can later draw some comparisons on their performance and results.

The paper is organized as follows. Section 2 and 3 reviews the techniques used in this work, covering zoning, the feature set used and MOEAs. Section 4 presents the methodology itself, while section 5 describes the tests and presents the results. Section 6 concludes this paper and outlines the future research.

2 Handwriting Recognition Issues

Our approach deals with two different aspects of handwriting recognition, zoning to select areas to extract local information from an image pattern, and feature extraction to actually extract information from each zone. This section briefly describe these two issues.

2.1 Zoning

A usual method to improve the recognition capability of handwriting recognition systems is through zoning [4]. Zoning is a method for local information analysis on partitions of a given pattern. The elements on such a partition are used to identify the position in which features of the pattern are found. Zones can be defined in portions of equal size, or non-proportionally, where zones may not cover the entire pattern space and may as well overlap. We have previously worked with handwriten digits recognition using the zoning strategy depicted in Figure 1. This strategy was defined by a human-expert and has been used on many experiments using handwriten digits.

Zoning strategy is usually defined by human experts using domain knowledge. We propose in this paper a self-adaptative methodology to define the zoning strategy with m non-overlapping zones and an acceptable error rate, with no need of human intervention during the search stage.

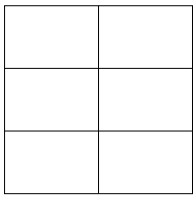


Figure 1. Zoning strategy

2.2 Features

The feature set used in our experiments is composed of a mixture of concavity and contour information. Thirteen measures of concavities, an histogram of contour directions (8-Freeman directions), and the number of black pixels are extracted from each zone of the image. In this way, the zoning strategy presented in Figure 1 will produce a feature vector of 132 components (22 x 6). More details can be found in [7].

3 Multi-Objective Evolutionary Algorithms

A MOEA is a search method based on Darwin's evolutionary theory aplied to a population of possible solutions. Here we will focus on MOEAs based on genetic algorithms – GAs [5]. In these algorithms, a population of candidate solutions goes through genetic operators such as selection, cross-over (also known as mating) and mutation, which creates an offspring population hoping that it is better than the parent population. When working with GAs there is a fitness function to evaluate the quality of a given solution, which evaluates how good the solution is when compared to the objectives to optimize. This poses a problem on most real world problems as they do not have only one objective to optimize, so it is necessary to compose those objectives into a single function, usually using a weight vector, to allow the algorithm to associate fitness values to solutions. While this technique works there is one issue to concern: objectives may be conflicting and domain knowledge is mandatory to resolve them and to make them directly comparable.

This lead to research on MOEAs, based on GAs, to solve multi-objective optimization problems. In such a case, instead of assigning a fitness criteria to individuals, they are evaluated by non-dominance and by spatial distribution, and the result is not one best solution, but a set of non-dominated solutions evenly spaced, which represents the best configurations for the many objectives being optimized. On a bidimensional search space (two objectives), such set is known as the *Pareto-front*. Deb wrote a comprehensive book on the subject [2], presenting many algorithms and techniques to evaluate their performance.

For our research we have chosen the Controlled Elitist NSGA [3], based on previous experiments with many algorithms on a set of standard problems [10]. Based on the well known NSGA and NSGA-II algorithms, the controlled elitist NSGA features more diversity on the population than other algorithms, which is desirable to allow the algorithm to explore better the search space on some difficult problems.

The idea behind this algorithm is inherited from the NSGA-II algorithm. A parent population goes through cross-over and an offspring population is created, with the same number of individuals. Those two populations are merged and sorted, first by non-dominance criteria and then each non-dominated level is sorted by crowding distance. This second measure indicates the quality of each individual releated to the spatial distribution on the non-dominated front. The objective of MOEAs is to find solutions as close as possible to the true non-dominated set, and to find them covering the entire space of this set. As we usually work with finite populations, we can only cover a portion of this space, so solutions must be evenly distributed to attain this objective. The crowding distance ensures that niches featuring many solutions that presents low diversity will have lower ranks when compared to isolated individuals, which introduces higher diversity.

Once the individuals on this merged population are sorted, we use a distribution scheme, usually the geometric distribution, to select the individuals from each non-dominated front to compose the next generation. This is done to help the algorithm to direct the search over the space, as the diversity introduced by genetic information of worse non-dominated levels may help the algorithm to converge on difficult search spaces. Using the geometric distribution with a $p\%$ distribution factor, we select $p\%$ individuals from the first non-dominated level for the next generation. We select the $p\%$ remaining individuals to complete the population from the second non-dominated level and so on. If a non-dominated level does not have enough individuals to be selected, the missing individuals are selected from the next level. Also, if the algorithm still needs more individuals for the next generation and has already gone through all levels, it starts over again from the first level.

This is different from the NSGA-II, where on a population of j individuals, it would select the best j individuals on the combined population. This method adds pressure to the convergence and may loose genetic information that is not likely to be introduced again. This low pressure towards convergence approach on the controlled elitist NSGA is adequate to our problem, as we do not know the search space or if there are discontinuities on the non-dominated set.

4 Proposed Methodology

When applying GA or MOEA to solve a problem, the optimization algorithm will not change, but rather the way each individual is coded to represent the solution. Also, to speed-up the optimization process, we used a Beowulf cluster, so our MOEA is actualy a distributed MOEA – DMOEA. To avoid inserting specific considerations that arise when using true parallel evolutionary algorithms [1], we used a master-slave approach, which does not change the algorithm behaviour, but allows us to achieve the same results as with a single processor but in a shorter time.

4.1 Individual Coding and Evaluation

The zoning strategies we are looking for must provide both acceptable error rates and use a small set of features. The first objective is mandatory to use a given strategy later into a real system, while the second objective is related to the higher generalization power of smaller feature sets – as the number of features is fixed for each zone, hence to reduce the number of features we have to reduce the number of zones on the zoning strategy.

Intuitively, these two objectives translate directly as the objectives functions to optimize during the MOEA search, and they conflict with each other. With the objective functions defined, we procede to define the individual coding. For this experiment, we defined the zones based on fixed position divisions that can be turned on and off, based on the template in Figure 2.

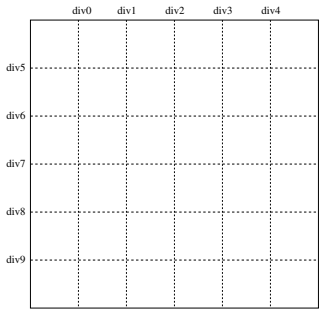


Figure 2. Individual coding template

Since each division has two states, we can define them based on a simple bit, which led us to a 10 bits string to code an individual, where each bit indicate whether the division is on or off. This string describes 1024 different possible zoning configurations that our algorithm will search and is presented in Figure 3.

To evaluate the individual's error rate, we use a Nearest Neighbor – NN – classifier. While this classifier is slow when compared to a neural network and other approaches on the classification phase, it does not require training for each different zoning strategy, once the features for a given strategy are extracted we can evaluate the individual's error rate using Equation 1, where $n_{correct}$ is the number or correct classifications and $n_{validation}$ is the size of the validation database. The number of zones of a given coding can be calculate by Equation 2, where div_x is a bit from the coding string.

div0	div1	div2	div3	div4	div5	div6	div7	div8	div9

Figure 3. Individual gene string

$$f_{error} = 1 - \frac{n_{correct}}{n_{validation}} \quad (1)$$

$$f_{zones} = (1 + \sum_{i=0}^{4} div_i) \times (1 + \sum_{j=5}^{9} div_j) \quad (2)$$

4.2 Cluster Topology

In this experiment we used a Beowulf cluster based on the MPI library, a well known library for the development of parallel processing applications based on message passing. The master-slave approach implementation is straightforward, the master node is responsible for all genetic operations and slave nodes are responsible to evaluate the individual's objective functions (Figure 4). To avoid wasting processing power on the cluster, the master node is also started on as a slave node, this way we have 7 slaves and one master on 7 physical nodes. Each node is a PC based on a Athlon processor running at 1.1GHz with 512MB of RAM.

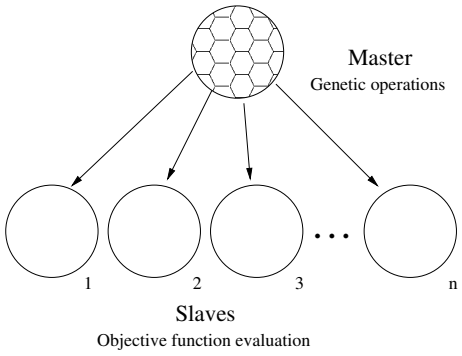


Figure 4. Master-slave topology (n slaves)

5 Experiments

To assess the proposed approach, we have used a random subset from the NIST SD-19 hsf-0123 handwriten dig-

its database with 50,000 observations for the training set, and another 10,000 for the validation set to evaluate the individual's error rate. To validate the zoning strategies found, we trained a neural network to compare with a zoning strategy defined previously [7].

5.1 MOEA Configuration

The MOEA being used, the controlled elitist NSGA, requires some configuration parameters, and they are defined for this experiment as follows:

- Number of individuals: 20
- Number of Generations: 25
- Single-point cross-over with 100% probability
- Bitwise Mutation: 0.1%
- Geometric distribution at 70%
- Crowding distance sorting
- Random initial population

To determine the population size and number of generations we considered the search space size, which features 1024 different possibilities, so we limited the algorithm to to explore at most 500 different possibilities (25×20). The actual number of different configurations explored will be smaller due to elitism and to redundant individuals generated by selection and cross-over when the algorithm converges towards the Pareto-front. Another parameter worth mentioning is the mutation rate, which was defined as $p_m = \frac{1}{L}$, where L is the individual coding string length (10 on our experiment). This definition for the mutation rate was based on earlier experiments with GAs and MOEAs.

5.2 Results and Discussion

During the experiment, the population converged by the 12th generation, which is explained by the size of the search space and confirms the choice for the population size and number of generations. Figure 5 shows the evolution of the population found during the experiment. The most important to note are the non-dominated solutions, which present the best solutions found by the algorithm.

Figures 6a to 6e show the zoning strategies found by our methodology that features the best trade-off between the number of zones and error rate. The baseline for comparison, designed by an human expert, depicted in Figure 6f, features 6 zones with an error rate of 5.32% and our methodology was able to find strategies that provided better results on the NN classifier with the same number of zones

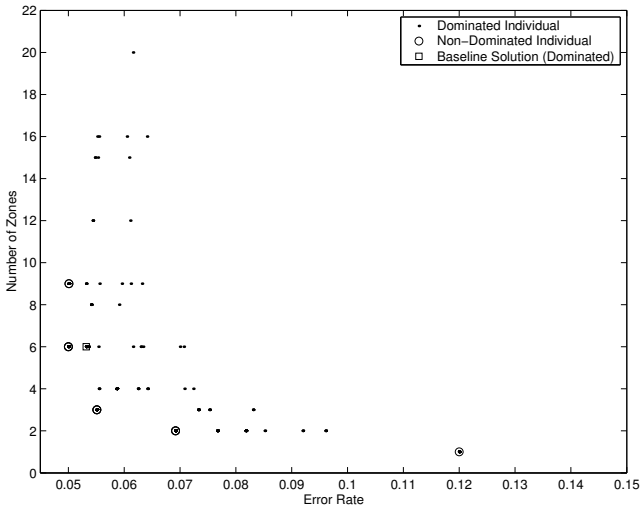


Figure 5. Experiment results

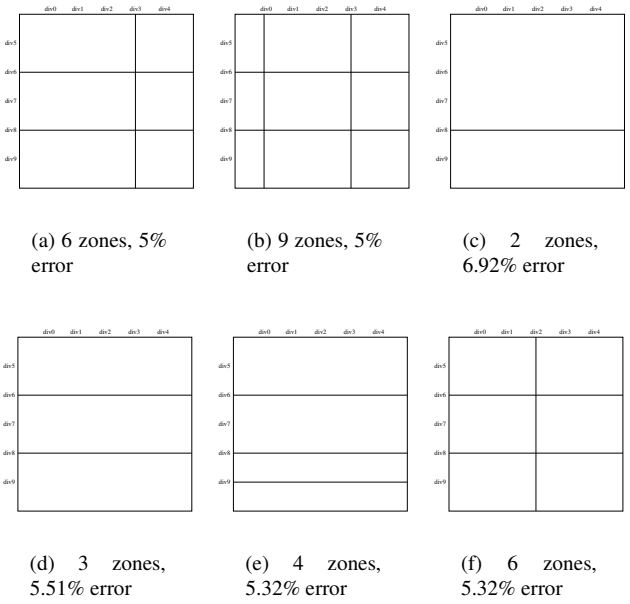
(Figure 6a), and a the same error rate with less zones (Figure 6e).

These results features diversity and may allow a human expert to choose the best trade-off between the number of features and error rate to select a zoning strategy and use it on a handwriting recognition application. To validate the methodology's generalization power we trained a neural network with the same databases and using another database for testing, with 10,000 digits from the hsf-7 database. This is demonstrated on Table 1, which shows the direct comparison between the error rates on the NN classifier and the neural network with the test database, where the non-dominance relation remains true and the error rate of the NN classifier is demonstrated to be effective on the zoning design task.

Strategy	*NN*	*Neural Network*
Figure 6a	5%	1.88%
Figure 6b	5%	1.85%
Figure 6c	6.92%	3.25%
Figure 6d	5.51%	2%
Figure 6e	5.32%	1.82%
Figure 6f	5.32%	1.97%

Table 1. Error rates

The results indicates the overal performance of our approach, leaving the path open for further experiments. The approach presented in [6] used a Beowulf cluster with 19 PCs based on 1.2GHz Athlon processors and required 4 weeks to complete the experiment, using a database with 3,750 observations. Our approach has been tested with a smaller cluster composed of 7 PCs and completed the ex-



(a) 6 zones, 5% error

(b) 9 zones, 5% error

(c) 2 zones, 6.92% error

(d) 3 zones, 5.51% error

(e) 4 zones, 5.32% error

(f) 6 zones, 5.32% error

Figure 6. Zoning strategies comparison

periment cycle in nine days. As the experiment converged at the 12^{th} generation, our experiment time actually falls to 6 days.

6 Conclusions

We proposed a methodology to select suitable zoning strategies for handwriten recognition using MOEAs. Comprehensive experiments using the NIST SD19 handwriten digits database proved the feasibility of the methodology to find adequate zoning strategies, without the requirement of domain expert feedback during the process, as the self-adaptative mechanism inherent to evolutionary algorithms provides the means to evolve solutions to above average results. On the experiment, we were able to find solutions that performed better than the baseline system.

We have yet to test the methodology with a larger number of classes, as alpha-numeric databases, but the concept is suitable for these cases also and will be the subject of future researches. We plan on experiment the methodology with other individual coding strategies, which will allow us to compare the performance of the methodology, as well as the results found. We also plan on expanding the methodology by using other feature sets and choosing the most suitable set for each zone during the search, which will improve the capabilities to better represent the handwriten symbols.

References

[1] E. Cantú-Paz. *Efficient and Accurate Parallel Genetic Algorithms*. Kluwer Academic Publishers, Norwell, Massachussets 02061 USA, 2000.

[2] K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, LTD, Baffins Lane, Chichester, West Sussex, PO19 1UD ,England, 2001.

[3] K. Deb and T. Goel. Controlled elitist non-dominated sorting genetic algorithms for better convergence. In E. Zitzler, K. Deb, L. Thiele, C. A. C. Coello, and D. Corne, editors, *Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization (EMO 2001)*, volume 1993 of *Lecture Notes in Computer Science*, pages 67–81, Berlin, 2001. Springer-Verlag.

[4] V. di Lecce, G. Dimauro, S. Impedovo, G. Pirlo, and A. Salzo. ZoningDesign for Hand-Written Numeral Recognition. In *Proceedings of the Seventh international Workshop on Frontiers in Handwriting Recognition – IWFHR-7*, pages 583–588, Amsterdan, 2000. Nijmegenl: International Unipen Foundation.

[5] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison Wesley, New York, NY, USA, 1989.

[6] A. Lemieux, C. Gagné, and M. Parizeau. Genetical Engineering of Handwriting Representation. In *Proceedings of the Eigth international Workshop on Frontiers in Handwriting Recognition – IWFHR-8*, pages 145–150, Ontario, Canada, 2002. IEEE Computer Society.

[7] L. S. Oliveira, R. Sabourin, F. Bortolozzi, and C. Y. Suen. Automatic recognition of handwritten numerical strings: A recognition and verification strategy. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(11):1438–1454, 2002.

[8] ϕivind Dur trier, A. K. Jain, and T. Taxt. Feature Extraction Methods for Character recognition – A Survey. *Pattern Recognition*, 29(4):641–662, 1996.

[9] A. Teredesai, J. Park, and V. Govindaraju. Active Handwritten Character Recognition using Genetic Programming. In *Proceedings of the European Conference on Genetic Programming – EuroGP*, 2001.

[10] E. Zitler, K. Deb, and L. Thiele. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation Journal*, 2(8):125–148, 2000.