An Online EHW Pattern Recognition System Applied to Face Image Recognition

Kyrre Glette[1], Jim Torresen[1], and Moritoshi Yasunaga[2]

[1] University of Oslo, Department of Informatics,
P.O. Box 1080 Blindern, 0316 Oslo, Norway,
{kyrrehg,jimtoer}@ifi.uio.no
[2] University of Tsukuba, Graduate School of Systems and Information Engineering,
1-1-1 Ten-ou-dai, Tsukuba, Ibaraki, Japan,
yasunaga@cs.tsukuba.ac.jp

Abstract. An evolvable hardware (EHW) architecture for high-speed pattern recognition has been proposed. For a complex face image recognition task, the system demonstrates (in simulation) an accuracy of 96.25% which is better than previously proposed EHW architectures. In contrast to previous approaches, this architecture is designed for online evolution. Incremental evolution and high level modules have been utilized in order to make the evolution feasible.

1 Introduction

Image recognition systems requiring a low recognition latency or high throughput could benefit from a hardware implementation. Furthermore, if the systems are applied in time-varying environments, and thus need adaptability, online evolvable hardware (EHW) would seem to be a promising approach [1].

One approach to online reconfigurability is the Virtual Reconfigurable Circuit (VRC) method proposed by Sekanina in [2]. This method does not change the bitstream to the FPGA itself, rather it changes the register values of a circuit already implemented on the FPGA, and obtains virtual reconfigurability. This approach has a speed advantage over reconfiguring the FPGA itself, and it is also more feasible because of proprietary formats preventing direct FPGA bitstream manipulation. However, the method requires much logic resources.

Experiments on image recognition by EHW were first reported by Iwata et al in [3]. A field programmable logic array (FPLA) device was utilized for recognition of three different patterns from black and white input images of 8x8 pixels. An EHW road image recognition system has been proposed in [4]. A gate array structure was used for categorizing black and white input images with a resolution of 8x4 pixels. Incremental evolution was applied in order to increase the evolvabiliy.

A speed limit sign recognition system has been proposed in [5]. The architecture employed a column of AND gates followed by a column of OR gates, and then a selector unit. A maximum detector then made it possible to decide

a speed limit from 6 categories. Incremental evolution was applied in two ways: each subsystem was first evolved separately, and then in a second step the subsystems were assembled and the selector units were evolved. The input images were black and white and had a resolution of 7x5 pixels.

An EHW face image classifier system, LoDETT, has been presented by Yasunaga et al. [6]. This system is capable of classifying large input vectors into several categories. For a face image recognition task, the input images had a resolution of 8x8 pixels of 8-bit grayscale values, belonging to 40 different categories. In this architecture, the classifier function is directly coded in large AND gates. The classification is based on detecting the category with the highest number of activated AND gates. Incremental evolution is utilized for this system too, where each module for detecting a category is evolved separately. The average recognition accuracy is 94.7% However, evolution is performed *offline* and the final system is synthesized. This approach gives rapid classification in a compact circuit, but lacks run-time reconfigurability.

The system we have developed earlier [7] addresses the reconfigurability by employing a VRC-like array of high-level functions. Online/on-chip evolution is attained, and therefore the system seems suited to applications with changes in the training set. However, the system is limited to recognizing one category out of ten possible input categories. The system uses the same image database as [6] with the same input resolution.

The architecture proposed in this paper expands to categorization of all 40 categories from the image database used in [6], while maintaining the on-line evolution features from [7]. A change in the architecture has been undertaken to accomodate for the recognition of multiple categories. While in LoDETT a large number of inputs to the AND gates could be optimized away during circuit synthesis, the run-time reconfiguration aspect of the following architecture has led to a different approach employing fewer elements.

A large amount of litterature exists on conventional face image recognition. A comprehensive survey can be found in [8]. Work on conventional hardware face recognition has been undertaken, based on the modular PCA method [9]. However, the recognition speed (11 ms) is still inferior to the LoDETT system.

The next section introduces the architecture of the evolvable hardware system. Aspects of evolution are discussed in section 3. Results from the experiments are given and discussed in sections 4. Finally, section 5 concludes the paper.

2 The EHW Architecture

The EHW architecture is implemented as a circuit whose behaviour and connections can be controlled through configuration registers. By writing the genome bitstream from the genetic algorithm to these registers, one obtains the phenotype circuit which can then be evaluated. This approach is related to the VRC technique, as well as to the architectures in our previous works [10,7].

2.1 System Overview

The classifier system consists of K category detection modules (CDMs), one for each category C_i to be classified – see figure 1. The input data to be classified are presented to each CDM concurrently on a common input bus. The CDM with the highest output will be detected by a maximum detector, and the number of this category will be output from the system. Alternatively, the system could also state the degree of certainty of a certain category by taking the output of the corresponding CDM and dividing by the maximum possible output. In this way, the system could also propose alternative categories in case of doubt.

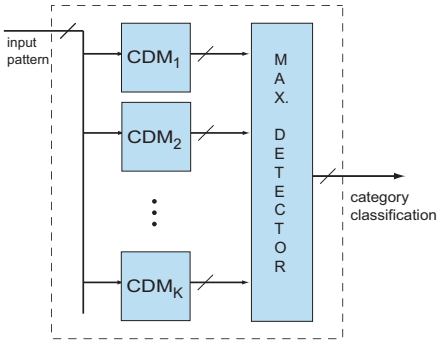

Fig. 1. EHW classifier system view. The pattern to be classified is input to all of the category detection modules.

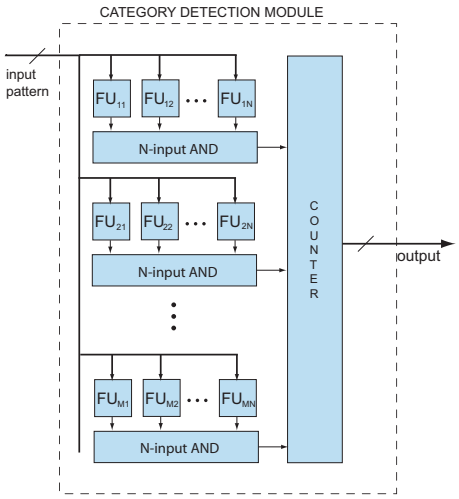

Fig. 2. Category detection module. N functional units are connected to an N-input AND gate.

2.2 Category Detection Module

Each CDM consists of M "rules" or functional unit (FU) rows. See figure 2. Each FU row consists of N FUs. The inputs to the circuit are passed on to the inputs of each FU. The 1-bit outputs from the FUs in a row are fed into an N-input AND gate. This means that all outputs from the FUs must be 1 in order for a rule to be activated. The outputs from the AND gates are connected to an input counter which counts the number of activated FU rows.

2.3 Functional Unit

The FUs are the reconfigurable elements of the architecture. As seen in figure 3, each FU behavior is controlled by configuration lines connected to the configuration registers. Each FU has all input bits to the system available at its

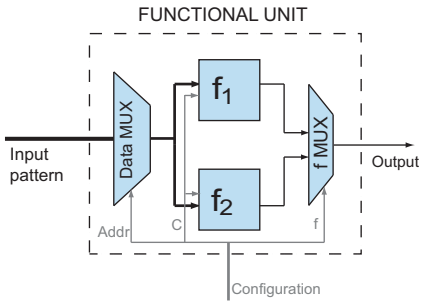

Fig. 3. Functional unit. The configuration lines are shown in gray. The data MUX selects which of the input data to feed to the functions f_1 and f_2. The constant C is given by the configuration lines. Finally, the f MUX selects which of the function results to output.

inputs, but only one data element (e.g. one byte) of these bits is chosen. One data element is thus *selected* from the input bits, depending on the configuration lines. This data is then fed to the available functions. The choice of functions for this application will be detailed in section 2.4. In addition, the unit is configured with a constant value, C. This value and the input byte are used by the function to compute the output from the unit.

The advantage of selecting which inputs to use, is that one is not required to connect to all inputs. A direct implementation of the LoDETT system [6] would have required, in the image recognition case, $N = 64$ FUs in a row. Our system typically uses $N = 6$ units. The rationale is that not all of the inputs are necessary for the pattern recognition. This is reflected in the *don't care*s evolved in [6].

2.4 Face Image Recognition

The pattern recognition system has been applied to face image recognition. The fitness of the face recognition system is based on the system's ability to recognize the correct person from a range of different face images. The images are taken from the AT&T Database of Faces (formerly "The ORL Database of Faces")[3] which contains 400 images divided into 40 people with 10 images each. For each person, images are taken with variations such as different facial expressions and head tilt. The original resolutions of the images were 92x112 pixels, 8-bit grayscale. In our experiment the images were preprocessed by a downsampling to 8x8 pixels, 8-bit grayscale. This was done to reduce noise and the number of inputs to the system. The input pattern to the system is then 64 pixels of 8 bits each (512 bits in total).

Based on the data elements of the input being 8-bit pixels, the functions available to the FU elements have been chosen to *greater than* and *less than*. Through experiments these functions have shown to work well, and intuitively

[3] http://www.cl.cam.ac.uk/Research/DTG/attarchive/facedatabase.html

this allows for detecting dark and bright spots. Combined use of these functions for the same pixel makes it possibe to define an intensity range. The constant is also 8 bits, and the input is then compared to this value to give *true* or *false* as output. This can be summarized as follows, with I being the selected input value, O the output, and C the constant value:

f	Description	Function
0	Greater than	$O = 1$ if $I > C$, else 0
1	Less than	$O = 1$ if $I < C$, else 0

3 Evolution

This section describes the evolutionary process. The genetic algorithm (GA) implemented for the experiments follows the Simple GA style [11]. The algorithm is written to be run on an embedded processor, such as the PowerPC 405 core in Xilinx Virtex-II Pro or better FPGAs [10]. Allowing the GA to run in software instead of implementing it in hardware gives an increased flexibility compared to a hardware implementation.

The GA associates a bit string (genome) with each individual in the population. For each individual, the EHW circuit is configured with the associated bit string, and training vectors are applied on the inputs. By reading back the outputs from the circuit, a fitness value can be calculated.

3.1 Genome

The encoding of each FU in the genome string is as follows:

Pixel address (6 bit)	Function (1 bit)	Constant (8 bit)

This gives a total of $B_{unit} = 15$ bits for each unit. The genome for one FU row is encoded as follows:

FU_1(15b)	FU_2(15b)	...	FU_N(15b)

The total amount of bits in the genome for one FU row is then, with $N = 8$, $B_{tot} = B_{unit} \times N = 15 \times 8 = 120$.

3.2 Incremental Evolution of the Category Detectors

Evolving the whole system in one run would give a very long genome, therefore an incremental approach is chosen. Each category detector CDM_i is evolved separately, since there is no interdependency between the different categories. This is also true for the FU rows each CDM consists of. Thus, the evolution can be performed on one FU row at a time. This significally reduces the genome size.

One then has the possibility of evolving CDM_i in M steps before proceeding to CDM_{i+1}. However, we evolve only *one* FU row in CDM_i before proceeding to CDM_{i+1}. This makes it possible to have a working system in K evolution runs (that is, $1/M$ of the total evolution time). While the recognition accuracy is reduced with only one FU row for each CDM, the system is operational and improves gradually as more FU rows are added for each CDM.

3.3 Fitness Function

A certain set of the available vectors, V_t, are used for training of the system, while the remaining, V_v, are used for verification after the evolution run.

Each row of FUs is fed with the training vectors ($v \in V_t$), and fitness is based on the row's ability to give a positive (1) output for vectors v belonging to its own category ($C_v = C_i$), while giving a negative (0) output for the rest of the vectors ($C_v \neq C_i$).

In the case of a positive output when $C_v = C_i$, the value A is added to the fitness sum. When $C_v \neq C_i$ and the row gives a negative output (value 0), 1 is added to the fitness sum. The other cases do not contribute to the fitness value. The fitness function F for a row can then be expressed in the following way, where o is the output of the FU row:

$$F = \sum_{v \in V_t} x_v \qquad \text{where } x_v = \begin{cases} A \times o \text{ if } C_v = C_i \\ 1 - o \text{ if } C_v \neq C_i \end{cases} \tag{1}$$

For the experiments, a value of $A = 64$ has been used. This emphasis on the positive matches for C_i has shown to speed up the evolution.

3.4 Evolution Parameters

For the evolution, a population size of 30 is used. Elitism is used, thus, the best individual from each generation is carried over to the next generation. The (single point) crossover rate is 0.9, thus the cloning rate is 0.1. A roulette wheel selection scheme is applied. Linear fitness scaling is used, with 6 expected copies of the best individual. The mutation rate is expressed as a probability for a certain number, n, of mutations on each genome. The probabilities are as follows:

n	1	2	3	4
$p(n)$	$\frac{7}{10}$	$\frac{1}{10}$	$\frac{1}{10}$	$\frac{1}{10}$

4 Results

This section presents the results of the experiments undertaken. The results are based on a software simulation of the EHW architecture.

4.1 Architecture parameters

The architecture parameters N and M, that is, the number of FUs in an FU row and the number of FU rows in a CDM, respectively, have been evaluated. As can be seen in figure 4, the number of generations required to evolve an FU row is dependent of the number of FUs in such a row.

Too few FUs makes it difficult for the FU row to distinguish between the right and wrong categories. Too many FUs give the same problem, as well as having the problem of a longer genome and thus a larger search space. We have

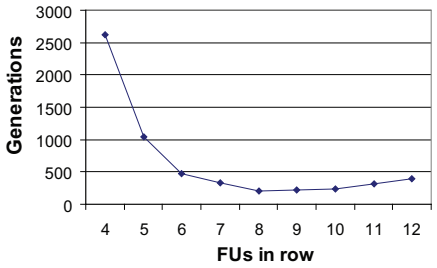

Fig. 4. Generations required to evolve rows of different number of FUs. Average over 5 evolution runs.

not seen any discernable connection between the number of FUs in an FU row and the recognition accuracy, as long as the row could be evolved to a maximum fitness value.

However, increasing the number of FU rows for a category leads to an increase in the recognition accuracy, as seen in figure 5. As the number of FU rows increases, so does the output resolution from each CDM. Each FU row is evolved from an initial random bitstream, which ensures a variation in the evolved FU rows. To draw a parallel to the system in [6], each FU row represents a kernel function. More FU rows give more kernel functions (with different centers) that the unknown pattern can fall into.

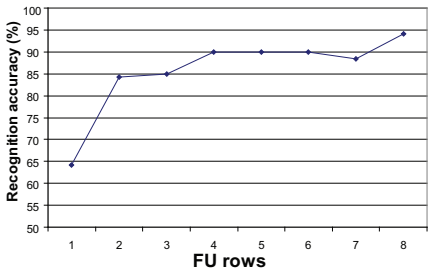

Fig. 5. Accuracy obtained for varying the number of FU rows. A fixed row size of 8 was used. Average over 3 runs.

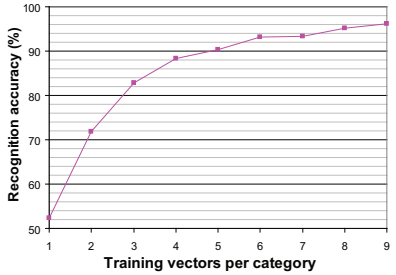

Fig. 6. Accuracy obtained for varying the number of training vectors per category, with $N = 6$ and $M = 10$.

4.2 Recognition Accuracy

10 evolution runs were conducted, each with a different selection of test vectors. That is, the 10% of the images which were used as test vectors were chosen differently in each evolution run. For $K = 40$, $M = 8$ and $N = 6$, an average recognition accuracy of 96.25% has been achieved. This result is slightly better than the average accuracy of 94.7% reported from the LoDETT system [6].

An experiment varying the number of training vectors has also been undertaken. 1 to 9 training vectors were used for each category. The rest were used as test vectors for calculating the accuracy. The results can be seen in figure 6. The results are competitive to traditional image recognition algorithms' results on the same dataset, such as Eigenfaces (around 90.0% for 8 training vectors per category) [12], or Fisherfaces (around 95.0% for 8 training vectors) [12], but other methods, such as SVM (98%) [13], perform better.

4.3 Evolution Speed

For $K = 40$, $M = 6$ and $N = 10$, the average number of generations (over 10 runs) required for each evolution run (that is, one FU row) is 219, thus an average of 52560 generations is required for the entire system. The average evolution time for the system is 140s on an Intel Xeon 5160 processor using 1 core. This gives an average of 0.6s for one FU row, or 23.3s for 40 rows (the time before the system is operational). It is expected that a hardware implementation will yield lower training times, as the evaluation time for each individual will be reduced.

4.4 Hardware Implementation

A preliminary implementation of an FU row has been synthesized for an FPGA in order to achieve an impression of resource usage. When synthesized for a Xilinx XC2VP30, 8 FU rows of 6 FUs (for one CDM) uses 328 slices, that is, 2% of the device. In this case, the data selector MUX in the FU is implemented by using time multiplexing, since directly implementing a 64x8-bit multiplexer for each FU requires many resources in the FPGA. The downside of time multiplexing is that each of the 64 pixels must be present on the inputs for a clock cycle before the classification can be made. With an estimate of maximum 10 cycles needed for the bit counter and the maximum detector, the system would require 74 clock cycles in order to classify a pattern. For a 100MHz system this would give more than 1M classifications per second, that is, less than 1μs for one image.

4.5 Discussion

The main improvement of this system over the LoDETT system is the aspect of on-line evolution. This is achieved by adapting a VRC-like architecture, allowing for quick reconfiguration. By selecting a subset of the input pixels for each image, the size of the circuit can be kept down. The drawback of this method is the extra time or resources needed for implementing the pixel selector MUXes. A positive side effect of the on-line adaptation is the increased recognition accuracy.

Real-time adaptation could be achieved by having evolution running on one separate FU row implemented on the same chip as the operational image recognition system. Thus, when there are changes in the training set (e.g. new samples are added), the FU rows can be re-evolved, one at a time, while the main system is operational using the currently best configuration. Since one FU row requires little hardware resources compared to the full system, little overhead is added.

A full system-on-chip hardware implementation is planned. The GA will run on an embedded processor in a Xilinx FPGA. Evolution speed as well as recognition speed should be measured. Furthermore, it would be interesting to improve the architecture in ways of dynamically adjusting the number of FUs and rows for each CDM, depending on its evolvability. Other architectural changes could also be considered, e.g. some kind of hierachical approaches, for increased recognition accuracy or improved resource usage.

Secondly, variations of the architecture should be tested on other pattern recognition problems. Since the main advantage of this architecture is its very high recognition speed, applications requiring high throughput should be suitable. The LoDETT system has been successfully applied to genome informatics and other applications [14,15]. It is expected that the proposed architecture also could perform well on similar problems, if suitable functions for the FUs are found.

5 Conclusions

An EHW architecture for a complex pattern recognition task has been proposed. The architecture supports run-time reconfiguration and is thus suitable for implementation in an on-chip evolution system. The architecture proposed utilizes data buses and higher level functions in order to reduce the search space. In addition, evolution of the system follows an incremental approach. Short evolution time is needed for a basic working system to be operational. Increased generalisation can then be added. The classification accuracy has been shown to slightly better than earlier offline EHW approaches. The system seems suitable for applications requiring high speed and online adaptation to a changing training set.

Acknowledgment

The research is funded by the Research Council of Norway through the project *Biological-Inspired Design of Systems for Complex Real-World Applications* (proj. no. 160308/V30).

References

1. Yao, X., Higuchi, T.: Promises and challenges of evolvable hardware. In Higuchi, T., et al., eds.: Evolvable Systems: From Biology to Hardware. First International Conference, ICES 96. Volume 1259 of Lecture Notes in Computer Science. Springer-Verlag (1997) 55–78
2. Sekanina, L., Ruzicka, R.: Design of the special fast reconfigurable chip using common F PGA. In: Proc. of Design and Diagnostics of Electronic Circuits and Sy stems - IEEE DDECS'2000. (2000) 161–168

3. Iwata, M., Kajitani, I., Yamada, H., Iba, H., Higuchi, T.: A pattern recognition system using evolvable hardware. In: Proc. of Parallel Problem Solving from Nature IV (PPSN IV). Volume 1141 of Lecture Notes in Computer Science., Springer-Verlag (September 1996) 761–770
4. Torresen, J.: Scalable evolvable hardware applied to road image recognition. In et al., J.L., ed.: Proc. of the 2nd NASA/DoD Workshop on Evolvable Hardware, IEEE Computer Society, Silicon Valley, USA (July 2000) 245–252
5. Torresen, J., Bakke, W.J., Sekanina, L.: Recognizing speed limit sign numbers by evolvable hardware. In: Parallel Problem Solving from Nature. Volume 2004., Springer Verlag (2004) 682–691
6. Yasunaga, M., Nakamura, T., Yoshihara, I., Kim, J.: Genetic algorithm-based design methodology for pattern recognition hardware. In Miller, J., et al., eds.: Evolvable Systems: From Biology to Hardware. Third International Conference, ICES 2000. Volume 1801 of Lecture Notes in Computer Science., Springer-Verlag (2000) 264–273
7. Glette, K., Torresen, J., Yasunaga, M., Yamaguchi, Y.: On-chip evolution using a soft processor core applied to image recognition. In: Proc. of the First NASA /ESA Conference on Adaptive Hardware and Systems (AHS 2006), Los Alamitos, CA, USA, IEEE Computer Society (2006) 373–380
8. Zhao, W., Chellappa, R., Phillips, P.J., Rosenfeld, A.: Face recognition: A literature survey. ACM Comput. Surv. **35**(4) (2003) 399–458
9. Ngo, H., Gottumukkal, R., Asari, V.: A flexible and efficient hardware architecture for real-time face recognition based on eigenface. In: Proc. of IEEE Computer Society Annual Symposium on VLSI, IEEE (2005) 280–281
10. Glette, K., Torresen, J.: A flexible on-chip evolution system implemented on a Xilinx Virtex-II Pro device. In: Evolvable Systems: From Biology to Hardware. Sixth International Conference, ICES 2005. Volume 3637 of Lecture Notes in Computer Science. Springer-Verlag (2005) 66–75
11. Goldberg, D.: Genetic Algorithms in search, optimization, and machine learning. Addison–Wesley (1989)
12. Zhou, D., Yang, X.: Face recognition using enhanced fisher linear discriminant model with facial combined feature. In: PRICAI. Lecture Notes in Computer Science, Springer-Verlag (2004) 769–777
13. Kim, K., Kim, J., Jung, K.: Recognition of facial images using support vector machines. In: Proceedings of the 11th IEEE Signal Processing Workshop on Statistical Signal Processing, IEEE (2001) 468–471
14. Yasunaga, M., et al.: Gene finding using evolvable reasoning hardware. In A. Tyrrel, P.H., Torresen, J., eds.: Evolvable Systems: From Biology to Hardware. Fifth International Conference, ICES'03. Volume 2606 of Lecture Notes in Computer Science. Springer-Verlag (2003) 228–237
15. Yasunaga, M., Kim, J.H., Yoshihara, I.: The application of genetic algorithms to the design of reconfigurable reasoning vlsi chips. In: FPGA '00: Proceedings of the 2000 ACM/SIGDA eighth international symposium on Field programmable gate arrays, New York, NY, USA, ACM Press (2000) 116–125