JAIST Repository
https://dspace.jaist.ac.jp/

Title	An analog VLSI implementation of one-class support vector machine for multiclass classification of highly dimensional vectors
Author(s)	Zhang, Renyuan; Kaneko, Mineo; Shibata, Tadashi
Citation	Japanese Journal of Applied Physics, 53(4S): 04EE03-1-04EE03-8
Issue Date	2014-02-28
Type	Journal Article
Text version	author
URL	http://hdl.handle.net/10119/12155
Rights	This is the author's version of the work. It is posted here by permission of The Japan Society of Applied Physics. Copyright (C) 2014 The Japan Society of Applied Physics. Renyuan Zhang, Mineo Kaneko and Tadashi Shibata, Japanese Journal of Applied Physics, 53(4S), 2014, 04EE03-1-04EE03-8. http://dx.doi.org/10.7567/JJAP.53.04EE03
Description	

JAIST
JAPAN
ADVANCED INSTITUTE OF
SCIENCE AND TECHNOLOGY

Japan Advanced Institute of Science and Technology

An Analog VLSI Implementation of One-Class Support Vector Machine for Multiclass Classification of Highly Dimensional Vectors

Renyuan Zhang[1*], Mineo Kaneko[1], and Tadashi Shibata[2]

[1]*School of Information Science, Japan Advanced Institute of Science and Technology*
Nomi, Ishikawa 923-1292, Japan
[2]*Center for Innovative Integrated Electronics Systems, Tohoku University*
Sendai, Miyagi 980-0845, Japan
[*]*E-mail: rzhang@jaist.ac.jp*

A one-class support vector machine (OC-SVM) is implemented using an on-chip-trainable analog VLSI processor. The one-class classification of highly dimensional sample vectors can be solved with this analog processor. Since the OC-SVM learning mechanism is complicated, a special solution scheme for the learning operation is proposed on the basis of analog computational circuitries and a fully parallel architecture. In this manner, the built VLSI processor achieves a high learning speed and a compact chip area at the same time. By combining multiple OC-SVM processors, multiclass recognition can be implemented with an arbitrary number of classes. The proof-of-concept chip is fabricated for the recognition of 64-dimensional vectors representing real image patterns. Three OC-SVM processors are combined for three classes of samples, where all the on-chip learning operations are accomplished within $0.6\ \mu s$. From the measurement results, all the test patterns are correctly recognized or rejected by the recognition system built.

1. Introduction

The support vector machine (SVM) has shown excellent performance in solving the pattern recognition problem of highly dimensional sample vectors, particularly when a Gaussian function kernel is employed.[1)] It has been widely applied in many real-world tasks of machine learning and artificial intelligence such as audio, image, and video processing even.[2–4)] For the efficient use of the SVM mechanism, many attempts were made to implement the SVM algorithm using VLSI processors in a distributed and parallel manner.[5,6)] On the other hand, the Gaussian kernel function is very expensive in silicon owing to its complexity. Several analog VLSI implementations of an on-chip trainable SVM have been developed with a reduced complexity of circuitry.[7,8)] As a powerful machine that learns algorithms, the SVM always has a time-consuming learning operation. In our previous work, a special fully parallel VLSI architecture was proposed to

improve the on-chip learning speed of the SVM.[9)]

Traditionally, the SVM algorithm was developed for solving binary classification problems. Two classes of learning samples are needed for binary SVM classification. A complicated mechanism was introduced to solve multiclass classification problems.[10–12)] In real-world applications, various numbers of classes are required; even only a single class of learning samples is available in some applications. To solve these problems, the one-class classification was developed as an extension of the SVM theory,[13–17)] which is named one-class support vector machine (OC-SVM). In these previous works, much advanced performance and various properties of the OC-SVM algorithm have been shown by demonstrating two-dimensional pattern vectors. In most practical tasks, a high dimensionality of vectors is required to represent the feature information of nature (image features for instance). Thus, several previous works showed the highly dimensional OC-SVM by software programs, and even achieved noticeable performance in solving practical face-tracking problems.[18,19)]

However, the learning operation of the OC-SVM algorithm requires iterative matrix computations. Therefore, software implementations of OC-SVM learning are very time-consuming, even if some advanced mathematical methods are employed.[15)] Conventional hardware implementation strategies can hardly be applied for OC-SVM problems owing to the matrix computations. Regarding highly dimensional pattern recognition, the problem becomes even more complicated. A previous work has shown the FPGA implementation of the OC-SVM.[20)] Unfortunately, its learning operation is pre-processed off-chip. In fact, a fully parallel analog VLSI could be one of the solutions for the on-chip learning of the OC-SVM. The feasibility of this strategy was analyzed and verified in our early study.[21)]

The purpose of this work is to develop a VLSI system for the multiclass recognition of highly dimensional pattern vectors. The OC-SVM algorithm is employed for learning every class of samples. On the basis of our previously developed analog fully parallel architecture, a special solution scheme of the OC-SVM problem is proposed. An analog VLSI processor is designed to implement the OC-SVM on-chip learning (and also recognition) operation for each class of learning samples, individually. The proof-of-concept chip is fabricated for the recognition of 64-dimensional sample vectors. By combining three chips, a multiclass classification system for recognizing the feature vectors from three (for demonstration but not limited to) classes of object images is demonstrated. All the learning operations are accomplished within 0.6 μs. From the chip measurement

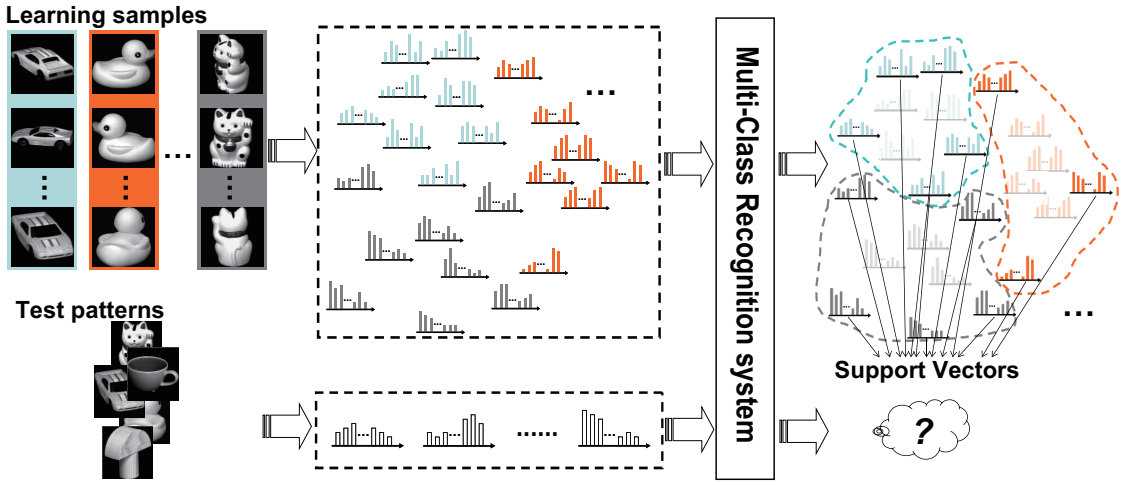


Fig. 1. Multiclass classification task for images employing the OC-SVM mechanism.

results, all the test patterns are correctly recognized or rejected by the system we built.

The rest of this paper is organized as follows: In Sect. 2, the target application of this work and a special solution scheme of the OC-SVM algorithm are presented on the basis of a fully parallel architecture. In Sect. 3, the analog OC-SVM processor and its operational principle are described. The experimental verifications of the multiclass recognition system are shown in Sect. 4 by considering both simulation and measurement results. Finally, conclusions are given in Sect. 5.

2. Algorithm

2.1 Target application of this work

It has been reported that the current-voltage characteristics of analog VLSI devices enable efficient computations in some complicated functions.[22,23] These reports indicate the potential of analog circuitries to offer a computational scheme of low power, high speed, compact size, and high parallelism. On the other hand, it is widely agreed that the analog computation suffers from an inaccuracy problem due to noise and variations among analog devices. However, absolute accuracy is not required in so-called soft-computing tasks such as pattern recognition.[24] The reasonable use of analog VLSI processors is helpful for performing a complex real-world task.[25] The target application of this work is image recognition using multiclass samples. In a conventional SVM algorithm, the learning operation is carried out using two classes of learning samples. Thus, the SVM is one of the so-called binary classification algorithms. However, multiclass recognition is always needed in real-world applications. The OC-SVM algorithm

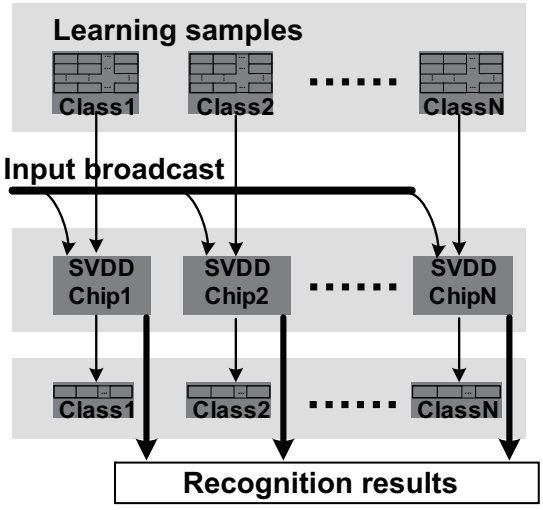
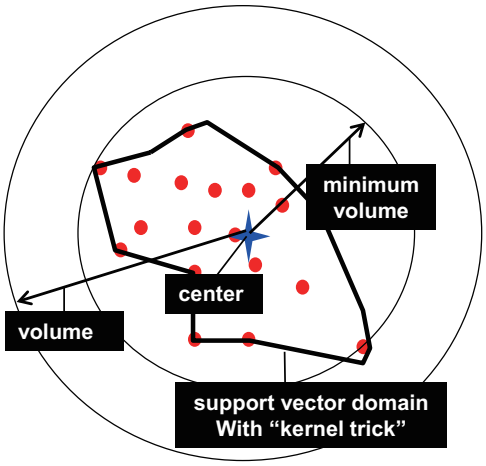


Fig. 2. Organization of multiclass recognition system employing OC-SVM algorithm.

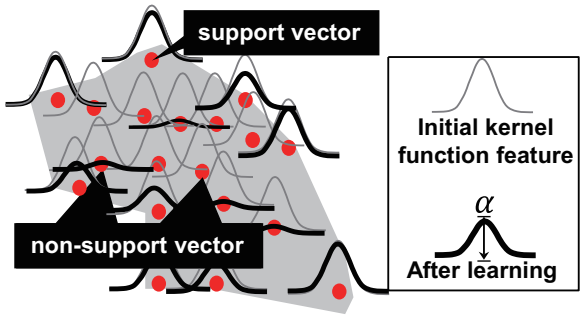
is expected to solve this kind of problem. A single class of pattern vectors is used as learning samples. After the learning operation, the data domain of each class is described through the OC-SVM mechanism by "support vectors" (SVs). The combination of many OC-SVM processors enables multiclass recognition. Figure 1 shows a multiclass classification task for images employing the OC-SVM algorithm. Many labeled image patterns are input into the multiclass recognition system as learning samples. According to the on-chip learning results, the test patterns are recognized or rejected.

In fact, each class of samples and its learning operation are independent of each other. Therefore, the number of classes can be freely increased or decreased depending on the application demand. One VLSI learning chip is applied to the on-chip learning of one class. During the recognition session, the test pattern is broadcasted to all the chips, as shown in Fig. 2. The result is output by this system in real time. In this manner, high-performance multiclass recognition is achieved.

There are several benefits of using the OC-SVM algorithm in solving image recognition problems. By sharing some properties with the binary SVM, the OC-SVM algorithm can obtain an accurate classification boundary through a markedly reduced number of samples. On the other hand, the number of classes can be freely expanded. Furthermore, the volume of data domains can be reasonably given by the learning operation depending on the application demand.



(a)



(b)

Fig. 3. Principle of OC-SVM algorithm: (a) the target of learning is to find a compact and flexible domain boundary for the given learning samples; (b) this boundary is described by several "support vectors" with different peak heights of the Gaussian-kernel function feature.

2.2 OC-SVM algorithm

The OC-SVM is a classification algorithm developed to solve the recognition problem using only one class of samples.[13] Two sessions are involved in the OC-SVM theory. The task of the learning session is to describe the domain boundary of the given samples (labeled); during the recognition session, when unlabeled test patterns are input, it is necessary to "accept" or "reject" them depending on the obtained boundary (in the form of a decision function).

The basic target of OC-SVM learning is to find a compact and flexible domain boundary for the given learning samples, as shown in Fig. 3(a). Clearly, if this boundary is a sphere with a sufficiently large volume, all the samples can be included. Even

defining a compact sphere with a minimum volume to include all the samples is easy. However, considering most practical applications, a flexible description rather than a simple "sphere" is necessary. In this sense, the OC-SVM learning session with the so-called "kernel trick" is expected to solve this problem.

To classify the n-dimensional vectors $\mathbb{X}$s in the form of $\mathbb{X} = (x_1, x_2, \cdots, x_n)$, a set of learning samples $\{\mathbb{X}_i, 1 \leq i \leq N\}$ is given, where N is the number of samples.

At the starting point, we expect to find a sphere with a minimum volume R, containing all (or most of) the data objects, which is defined by

$$F(R, \mathbf{a}, \xi_i) = R^2 + C \sum_i \xi_i, \tag{1}$$

where $\mathbf{a}$ is the center of this sphere, ξ_i is a slack variable for error tolerance, and the parameter C gives a trade-off between simplicity (or the volume of the sphere) and the number of errors (number of target objects rejected). This function has to be minimized under the constraint

$$(\mathbb{X}_i - \mathbf{a})^{\mathbf{T}}(\mathbb{X}_i - \mathbf{a}) \leq R^2 + \xi_i, \tag{2}$$

By applying the theory developed for the original OC-SVM algorithm, and considering the Gaussian Kernel function $K(\mathbb{X}_i, \mathbb{X}_j) = exp(-\| \mathbb{X}_i - \mathbb{X}_j \|^2/\sigma)$, the task becomes the following quadratic programming (QP) problem:[14)]

$$\min L = 1 - \sum_i \alpha_i^2 - \sum_{i \neq j} \alpha_i \alpha_j K(\mathbb{X}_i, \mathbb{X}_j), \tag{3}$$

under the constraints

$$\sum_i \alpha_i = 1, \tag{4}$$

and $0 \leq \alpha_i \leq C$. When Lagrangian multipliers (αs) are obtained by solving this QP problem, they can be directly used in the test session for the recognition of test patterns.

During the test session, to determine whether a test object $\mathbb{Z}$ is within the sphere, the distance to the center of the sphere has to be calculated. When $(\mathbb{Z}-\mathbf{a})^{\mathbf{T}}(\mathbb{Z}-\mathbf{a}) \leq R^2$ is satisfied, the object $\mathbb{Z}$ is accepted. Expressing the sphere center in the form of the support vectors, the object $\mathbb{Z}$ is accepted when

$$1 - 2\sum_i \alpha_i K(\mathbb{Z}, \mathbb{X}_i) + \sum_{i,j} \alpha_i K(\mathbb{X}_i, \mathbb{X}_j) \leq R^2. \tag{5}$$

Theoretically, the distance from any support vector $\mathbb{X}_s$ to the center should be R. Thus,

the constant items in Eq. 5 could be observed using any support vector in the following form:

$$1 + \sum_{i,j} \alpha_i K(\mathbb{X}_i, \mathbb{X}_j) - R^2 = 2 \sum_i \alpha_i K(\mathbb{X}_s, \mathbb{X}_i). \tag{6}$$

In this manner, the object $\mathbb{Z}$ is accepted when the following condition is satisfied:

$$\sum_i \alpha_i K(\mathbb{Z}, \mathbb{X}_i) \geq \sum_i \alpha_i K(\mathbb{X}_s, \mathbb{X}_i). \tag{7}$$

In this work, the support vector with the largest alpha value is selected for observation.

2.3 Proposed scheme to solve the QP problem in OC-SVM algorithm

The core part of the OC-SVM algorithm is to solve the QP problem described by Eq. 3. When the complex kernel functions such as Gaussian function are employed, the process for solving this QP problem could be very time-consuming. Furthermore, since this problem includes the kernel function computations between each arbitrary pair of learning samples, a large number of costly complex matrix computations are required. In some reported works, various mathematical approaches such as sequential minimal optimization (SMO) were considered to reduce the number of these matrix computations. In our work, an iterative learning scheme for solving a specific QP problem is proposed by employing a large number of matrix computations. This scheme could be expensive for conventional implementations of software programs, but can be efficiently implemented using a fully parallel VLSI architecture.

To solve the QP problem, another Lagrangian function is constructed with the multiplier λ:

$$\begin{array}{c} \mathfrak{L} = L + \lambda(\sum_i \alpha_i - 1) \\ = 1 - \sum_i \alpha_i^2 - \sum_{i \neq j} \alpha_i \alpha_j K(\mathbb{X}_i, \mathbb{X}_j) + \lambda(\sum_i \alpha_i - 1) \end{array}. \tag{8}$$

Then, the following equations should be solved to minimize L:

$$\begin{cases} \frac{\partial \mathfrak{L}}{\partial \alpha_i} = 0 \\ \sum_i \alpha_i - 1 = 0 \end{cases} \tag{9}$$

The expansion of these equations is in the linear form as

$$\begin{cases} -2\alpha_1 - \sum_{j\neq 1}\alpha_j K(\mathbb{X}_1,\mathbb{X}_j) + \lambda = 0 \\ -2\alpha_2 - \sum_{j\neq 2}\alpha_j K(\mathbb{X}_2,\mathbb{X}_j) + \lambda = 0 \\ \vdots \\ -2\alpha_N - \sum_{j\neq N}\alpha_j K(\mathbb{X}_N,\mathbb{X}_j) + \lambda = 0 \\ \alpha_1 + \alpha_2 + \ldots + \alpha_N = 1 \end{cases}. \tag{10}$$

With the consideration of the Gaussian function kernel ($\alpha_i K(\mathbb{X}_i,\mathbb{X}_i) = \alpha_i$), these linear equations can be equivalently converted into

$$\begin{cases} -2\alpha_1 - \sum_{j\neq 1}\alpha_j K(\mathbb{X}_1,\mathbb{X}_j) + \lambda = 0 \\ -2\alpha_2 - \sum_{j\neq 2}\alpha_j K(\mathbb{X}_2,\mathbb{X}_j) + \lambda = 0 \\ \vdots \\ -2\alpha_N - \sum_{j\neq N}\alpha_j K(\mathbb{X}_N,\mathbb{X}_j) + \lambda = 0 \\ -\sum_i\sum_j \alpha_j K(\mathbb{X}_i,\mathbb{X}_j) + N\lambda = 1 \end{cases}. \tag{11}$$

The Jacobian iterative method is applied to solve these linear equations. Thus, the iterative updating rule is obtained as

$$\begin{cases} \alpha_i \leftarrow \frac{1}{2}(\lambda - \sum_{j\neq i}\alpha_j K_{ij}) \\ \lambda \leftarrow \frac{1}{N}(1 + \sum_i\sum_j \alpha_j K(\mathbb{X}_i,\mathbb{X}_j)) \end{cases}, \tag{12}$$

By considering the upper and lower boundaries of α and λ, the final updating rule becomes

$$\begin{cases} \alpha_i \leftarrow \max(0, \min(\frac{1}{2}(\lambda - \sum_{j\neq i}\alpha_j K_{ij}), C)) \\ \lambda \leftarrow \max(0, \frac{1}{N}(1 + \sum_i\sum_j \alpha_j K(\mathbb{X}_i,\mathbb{X}_j))) \end{cases}. \tag{13}$$

The trade-off parameter C should be set within an interval of $1/N \leq C \leq 1$ to ensure that the solution to the QP problem can be obtained.[13] As shown in Fig. 3(b), all the peak heights (α_is) of the Gaussian function are initialized to be the same at the beginning of learning. After learning, the alpha values are autonomously adjusted according to the learning rule. The sample vectors with nonzero alpha values are called "support vectors"; the others are called nonsupport vectors. These support vectors with Gaussian kernel functions describe a compact and flexible boundary of data domains.

clearly, a large scale of parallel matrix computation is needed to update λ by this method. This is the reason why a fully parallel architecture is considered in this work on the basis of an analog Gaussian-cell array.

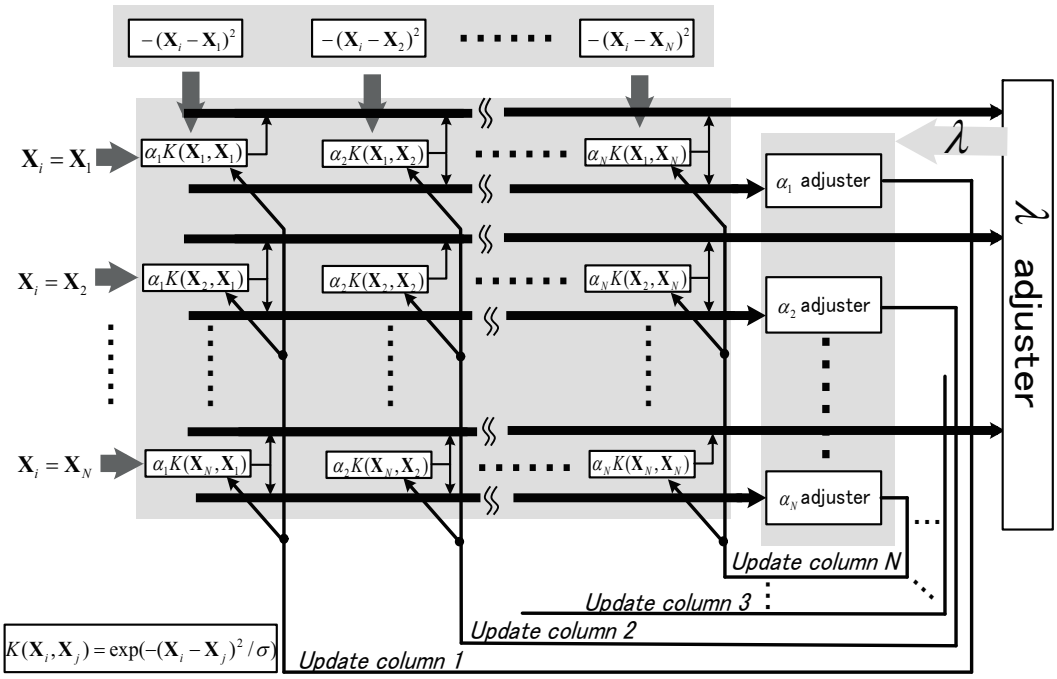


Fig. 4. Fully parallel on-chip learning OC-SVM processor.

3. Hardware implementation

A proof-of-concept processor is built to implement the learning operation of the OC-SVM algorithm in a 0.18 μm CMOS technology. The organization of the proposed processor is shown in Fig. 4 along with a micrograph of its chip in Fig. 5. The entire processor contains an analog Gaussian-cell array, a set of alpha adjusters, and a lambda adjuster. All the Gaussian kernel functions are carried out using the Gaussian-cell array in real time and in a fully parallel configuration. During the learning session, these function values are fed into the alpha and lambda adjusters; the updated alpha and lambda values are freely fed back to the Gaussian-cell array in the form of analog signals. In this manner, the learning operation proceeds autonomously and self-converges without any clock-based control. The chip-area-hungry part for highly dimensional Euclidean distance computations and the much smaller part for exponential computation are built separately. Only exponential computing circuits should be duplicated for a high degree of parallelism. Thus, a fully parallel array can be fabricated with a compact chip area.

During the test session, only Gaussian cells in the first row are used, and all the calculations for an object test are also carried out in parallel and in real time according to Eq. 7. As mentioned above, the support vector with the largest alpha value is selected as the reference in the decision equation. This selection mechanism is realized by the

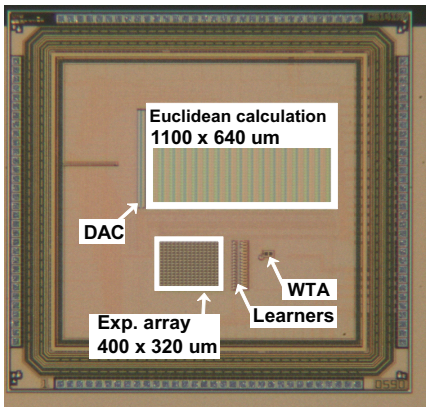


Fig. 5. Micrograph of fabricated OC-SVM learning chip.

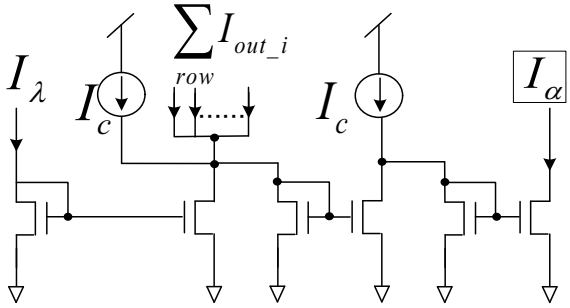


Fig. 6. Schematic of alpha adjuster circuit.

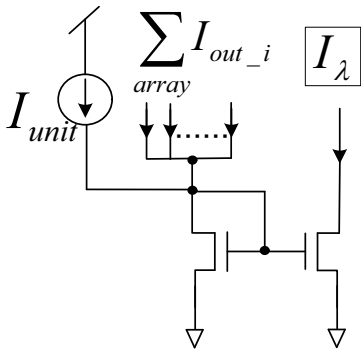


Fig. 7. Schematic of lambda adjuster circuit.

winner-take-all (WTA) module.

The analog Gaussian generation circuit, which was reported in our early work,[9] is employed to carry out the kernel function computation. Receiving two vectors in the form of the voltages $\mathbb{V}_i = (v_{i1}, v_{i2}, \cdots, v_{in})$ and $\mathbb{V}_j = (v_{j1}, v_{j2}, \cdots, v_{jn})$, the Gaussian function related to these two vectors can be computed as

$$I_{out} = \frac{I_\alpha}{2} e^{-\gamma ||\mathbb{V}_i - \mathbb{V}_j||^2}, \tag{14}$$

where

$$\gamma = \frac{K_n}{(V_{dd} - V_{bias} - |V_{thp}|)(\sqrt{K_p} + \sqrt{K_n})^2}. \tag{15}$$

Here, V_{thn} and V_{thp} are the threshold voltages of the n-type and p-type MOS transistors, respectively. The current I_α reflects the alpha value in Eq. 13, which is dynamically programmed by the alpha adjusters. The spread width of the Gaussian function feature is programmed by tuning the voltage signal V_{bias}.

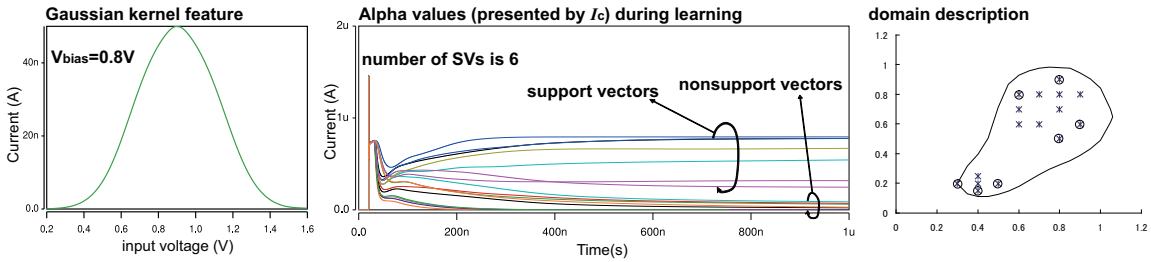
A current-mirror-based circuit is designed as the alpha adjuster, as shown in Fig. 6. By collecting the output current of the Gaussian cells in a specific row, the updated current I_α is output to the respective column of Gaussian-cell array according to the update rule in Eq. 13. The current source I_c is introduced to reflect the trade-off parameter C. The circuit schematic of the lambda adjuster is shown in Fig. 7. The lambda adjuster collects the output current from all the Gaussian kernel cells in the array, and the updated lambda value is fed into all the alpha adjusters. The current source I_{unit} reflects the constant factor "1" in Eq. 10. Therefore, the parameter C is represented by $C = I_c/I_{unit}$. All the elements in a fully parallel array are available and necessary for updating the lambda value. Thus, the proposed scheme to solve the OC-SVM problem particularly fits the fully parallel architecture rather than the series processing by software programs[18, 19] or the row parallel processing.[8]

4. Experiments

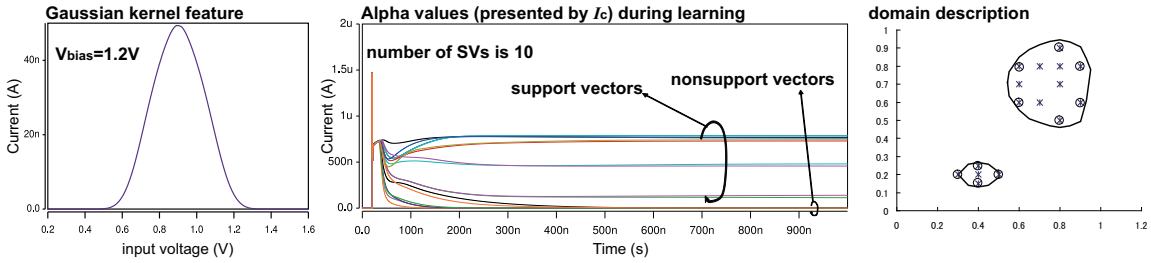
The circuit simulations and chip measurements are used to verify the performance of the OC-SVM processor developed. A single OC-SVM processor for 2-D sample vectors, a single OC-SVM processor for 64-D sample vectors, and a multiclass classification system employing multiple OC-SVM processors are considered.

4.1 Circuit simulations

To verify the correctness of the recognition results, circuit simulations are performed. Furthermore, some important inner signals (such as α_is) can be dynamically observed from the simulation results.



(a)



(b)

Fig. 8. Support vector domain description of a toy example with sixteen learning samples: (a) when a wide spread of Gaussian function feature is applied, a rough boundary is described by seven support vectors; (b) when a narrow spread of Gaussian function feature is applied, a finer boundary is described by ten support vectors.

4.1.1 2-D pattern recognition using single OC-SVM processor

A toy example with sixteen learning samples is set up to verify the performance of the proposed proof-of-concept processor. Each learning sample is represented by a set of voltage signals. For instance, the learning sample $\mathbb{X} = \{0.6, 0.6\}$ is represented by $\mathbb{V} = \{0.6V, 0.6V\}$. In this example, a Gaussian-cell array with 16×16 elements is constructed. The HSPICE simulation results and description boundaries are shown in Fig. 9 by applying different Gaussian function features. In this experiment, the parameter C is set at 0.08 by setting the currents I_{unit} and I_c at 10 and 0.8 μA, respectively. From the simulation results, there is no outlier with these configurations, and the learning operation is accomplished within about 0.6 μs. When a wide spread of Gaussian function feature is applied, a rough domain is described by seven support vectors, as shown in Fig . 8(a). Various schemes of data domain description can be considered as a benefit of the OC-SVM algorithm on the mathematical side. In this sense, the feature of the domain description should be adjustable by tuning some control signals on the circuit side. The

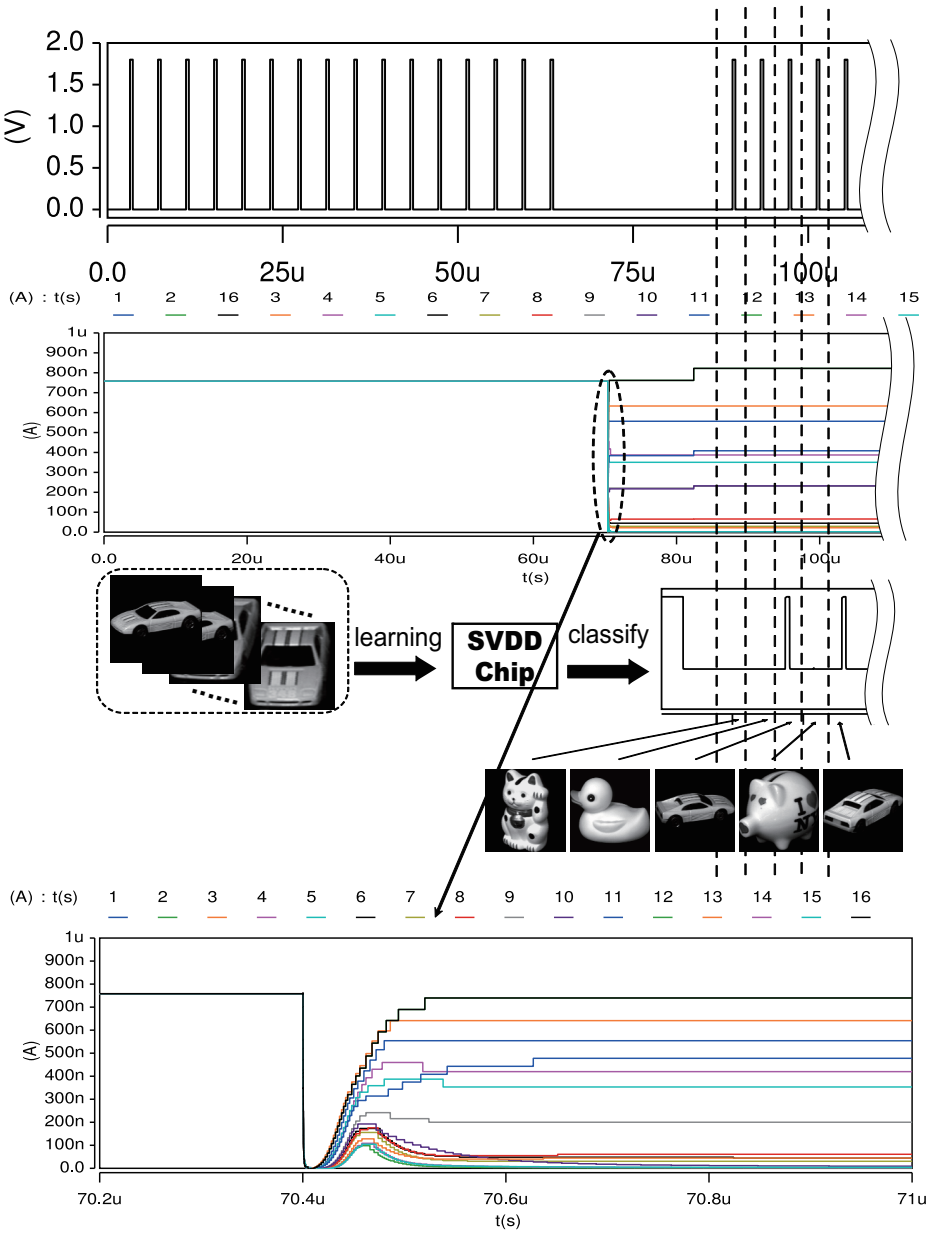


Fig. 9. Circuit simulation results of 64-D learning/classifying performance of single OC-SVM learning chip.

results in Fig. 8(b) reflect this variation, for instance. A finer domain description is obtained by applying a narrow spread of the Gaussian function feature, which is shown in Fig. 8(b). However, the number of support vectors increases in this case.

4.1.2 64-D pattern recognition employing single OC-SVM processor

For real-world applications, a multiclass classification system for image recognition is constructed using the OC-SVM learning chips. In the proof-of-concept system, three

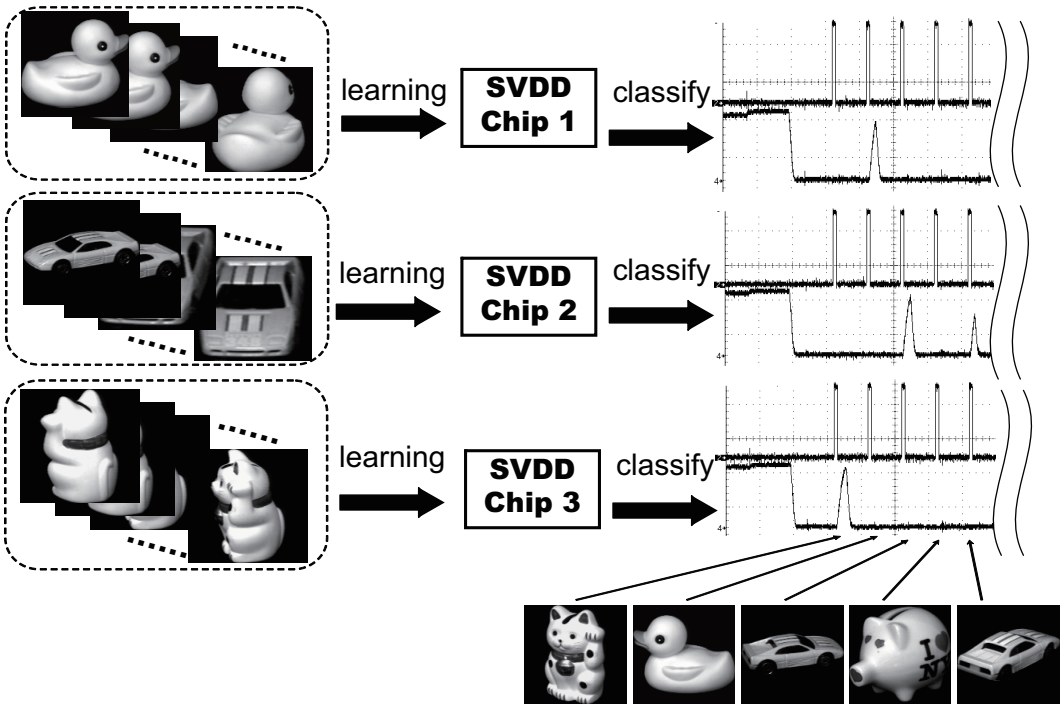


Fig. 10. Chip measurement results of an associative memory system employing OC-SVM learning processor: three chips are used as OC-SVM processors independently. The test patterns are broadcasted to all the chips.

fabricated chips are used as OC-SVM learners. Namely, three classes of images from the COIL-20 database[26] are employed as learning samples. The features of these images are extracted into 64-dimensional vectors using typical image-processing technology named the projected principal-edge distribution (PPED).[27] The recognition performance of a single OC-SVM learning chip is given in Fig. 9 by the circuit simulation results.

4.2 Multiclass classification system employing multiple OC-SVM processors

As has been reported previously,[10–12] binary classifiers can be used to construct a multiclass classification system using some hierarchical decision mechanisms. For instance, the one-against-others strategy is widely used to realize multiclass recognition using several sets of binary classifiers. However, a complex grouping and decision mechanism is necessary for this kind of effort. It is difficult to tackle a flexible number of classes particularly when the hardware implementation is considered. An important benefit of the OC-SVM recognition system we developed is the flexibility over the number of classes. The recognition of each class is processed independently. All the recognition results are output in parallel. Namely, it is easy to increase the number of classes by

employing more OC-SVM processors.

The chip measurement results for the proposed multiclass recognition system are shown in Fig. 10. Three chips are used as OC-SVM processors independently. In this manner, three classes of learning samples are used for data domain description. After the learning session, test patterns are broadcasted to all the chips. If the test pattern belongs to a specific class, the respective chip outputs a signal of high voltage. From this figure, some test patterns do not belong to any sample class. Thus, none of the output signals is of high voltage. Namely, the specific test patterns are rejected by all the existing classes of samples. Actually, this phenomenon reflects an important benefit of OC-SVM compared with the binary SVM. In the latter algorithm, any test pattern must be accepted by either class of samples. From the nature point of view, the "all-rejection" situation might appear in real-world recognition problems.

4.3 Comparisons

The OC-SVM algorithm has already been implemented by many different approaches using software programs, even hardware. The performances are compared between this work and the software implementations, as Table I shows. The proposed analog VLSI system of multiclass recognition clearly shows benefits in terms of both learning and recognition speed compared with the software implementations. On the other hand, the processing capacity (number of learning samples) is not superior owing to the inherent limitation of VLSI fabrications. However, the number of learning samples can be increased by employing reasonably more OC-SVM processing chips. Comparisons between this work and other hardware implementations are also given in Table II. The OC-SVM implementation is helpful for increasing the number of classes compared with the binary SVM. At the same time, the penalty on the chip area and power consumption is acceptable. In the referenced work with FPGA, effort to achieve low power was made on the digital side. The power consumption of this FPGA implementation (without consideration of the power consumption for the learning session) is still much higher than that achieved with our effort. Furthermore, the on-chip learning ability is an important consideration for the hardware implementations.

5. Conclusions

A multiclass recognition system was built for highly dimensional sample vectors. A fully parallel analog VLSI processor was designed to implement the one-class support vector

Table I. Performance comparisons between this work and the software implementations.

	Reference[15]	Reference[19]	This work
Implementation	Matlab	Matlab	Analog VLSI
Algorithm	OC-SVM	OC-SVM	OC-SVM
Kernel function	Gaussian	Gaussian	Gaussian
Max Number of samples	171	2468	16 (per chip)
Number of dimensions	$2 \sim 19$	57	$2 \sim 64$
Learning speed (vectors/s)	0.31 (19-D)	3.32	26.7×10^6
Recognition speed (vectors/s)	N/A	5×10^3	10^6 (for DAC)

Table II. Performance comparisons between this work and other hardware implementations.

	Reference[9]	Reference[20]	This work
Implementation	Analog VLSI	FPGA	Analog VLSI
Algorithm	Binary SVM	OC-SVM	OC-SVM
Learning parallelism	Fully parallel	Off-chip (pre-processed)	Fully parallel
Number of classes	2	Arbitrary	Arbitrary
Chip area	0.85 mm^2	Board	0.89 mm^2
Power	0.026 W	2.04 W	0.035 W

machine. Each class of samples is used for on-chip learning by a single analog processor, individually. The learning operation is autonomously accomplished within 0.6 μs. By combining multiple chips of OC-SVM processors, the recognition problem is solved for an arbitrary number of classes. To verify the performance of the system, three classes of 64-dimensional vectors representing image features were employed as learning samples. All the on-chip learning and recognition operations were performed by the recognition system with three OC-SVM processors. From the chip measurement results, all the test patterns were correctly recognized or rejected.

Acknowledgments

The authors would like to thank the VLSI Design and Education Center (VDEC) of the University of Tokyo in collaboration with Rohm Corporation, Toppan Printing Corporation, Synopsys, Inc., Cadence Design Systems, Inc., and Mentor Graphics, Inc.

References

1) V. Vapnik: *The Nature of Statistical Learning Theory* (Springer, New York, 1995) p. 23.
2) S. Cumani and P. Laface: IEEE Trans. Audio Speech Lang. Process. **20** (2012) 1585.
3) S. Moustakidis, G. Mallinis, N. Koutsias, J. B. Theocharis, and V. Petridis: IEEE Trans. Geosci. and Remote Sens. **50** (2012) 149.
4) Y. Liu and Y. F. Zheng: IEEE Trans. Signal Process. **55** (2007) 3272.
5) D. Anguita, A. Boni, and S. Ridella: IEEE Trans. Neural Networks **14** (2003) 993.
6) M. Shi and A. Bermak: IEEE Trans. VLSI Syst. **14** (2006) 962.
7) S.-Y. Peng, B. A. Minch, and P. E. Hasler: Proc. Int. Symp. Circuits Syst., 2008, p. 860.
8) K. Kang and T. Shibata: IEEE Trans. Circuits Syst. **57** (2010) 1513.
9) R. Zhang and T. Shibata: Jpn. J. Appl. Phys. **51** (2012) 04DE10.
10) F. F. Chamasemani and Y. P. Singh: Proc. IEEE Int. Conf. Bio-Inspired Computing: Theories and Applications, 2011, p. 351.
11) I. Abbasnejad, M. J. Zomorodian, and E. T. Yazdi: Proc. IEEE Int. Conf. Mechatronics and Machine Vision in Practice: Theories and Applications, 2012, p. 408.
12) Y. Wang: Proc. IEEE Int. Conf. MultiMedia and Information Technology, 2008, p. 15.
13) D. M. J. Tax, and R. P. W. Duin: Proc. European Symp. on Artificial Neural Networks, 1999, p. 251.
14) D. M. J. Tax, and R. P. W. Duin: Pattern Recognition Lett. **20** (1999) 1191.
15) H. Wang, G. Zhao, and H. Gu: Proc. IEEE Int. Conf. Natural Computation, 2010, p. 824.
16) J. Xu, J. Yao, and L. Ni: Proc. IEEE Int. Conf. Electronics, Communications and Control, 2011, p. 2050.
17) X. Huang and X. Chen: Proc. IEEE Global Congr. Intelligent Systems, 2009, p. 486.
18) L. Zhao, Y. Song, Y. Zhu, C. Zhang, and Y. Zheng: Proc. IEEE Control and Decision Conf., 2009, p. 5871.
19) L. Guo, L. Zhao, Y. Wu, Y. Li, G. Xu, and Q. Yan: IEEE Trans. Magn. **47** (2011)

3849.

20) R. A. Patil, G. Gupta, V. Sahula, and A. S. Mandal: Proc. IEEE Int. Conf. VLSI Design, 2012, p. 62.
21) R. Zhang and T. Shibata: Proc. IEEE Int. Workshop Cellular Nanoscale Networks and their Applications, 2012, p. 1.
22) K. H. Lundberg: IEEE Control Syst. Mag. **25** (2005) 22.
23) T. Delbruck.: IEEE Int. Joint Conf. Neural Networks, 1991, p. 475.
24) T.-W. Chen, C.-H Sun, H.-H. Su, S.-Y. Chien, D. Deguchi, I. Ide, and H. Murase: IEEE J. Emerging Sel. Top. in Circuits Syst., **1** (2011) 357.
25) P. Zhao, R. Zhang and T. Shibata: Lecture Notes in Computing Science (Springer, Heidelberg), 2012, Vol. 7267, p. 617.
26) S. A. Nene, S. K. Nayar, and H. Murase: Columbia Object Image Library (COIL-20), Tech. Rep. CUCS-005-96.
27) M. Yagi and T. Shibata: IEEE Trans. Neural Networks **14** (2003) 1144.