論 文
53D-8-5

Concurrent Support Vector Machine 프로세서

Concurrent Support Vector Machine Processor

魏 載 玕* · 李 鍾 浩**
(Jae Woo Wee · Chong Ho Lee)

Abstract - The CSVM(Current Support Vector Machine) that is a digital architecture performing all phases of recognition process including kernel computing, learning, and recall of SVM(Support Vector Machine) on a chip is proposed. Concurrent operation by parallel architecture of elements generates high speed and throughput. The classification problems of bio data having high dimension are solved fast and easily using the CSVM. Quadratic programming in original SVM learning algorithm is not suitable for hardware implementation, due to its complexity and large memory consumption. Hardware-friendly SVM learning algorithms, kernel adatron and kernel perceptron, are embedded on a chip. Experiments on fixed-point algorithm having quantization error are performed and their results are compared with floating-point algorithm. CSVM implemented on FPGA chip generates fast and accurate results on high dimensional cancer data.

Key Words : SVM(Support Vector Machine), FPGA, Gene Expression Data

1. 서 론

최근 많은 차원의 데이터를 다루는 바이오인포매틱스(bioinformatics) 분야의 문제가 대두되고 있으나 고차원의 데이터를 다루는 데에 어려움을 많이 겪고 있다. DNA 마이크로어레이(microarray) 칩을 통해 환자의 유전자 발현양상을 측정하여 질병의 유형을 분류하는 데에 여러 종류의 기계 학습 방법이 사용되고 있다. 그 방법 중에 1990년대 중반 Vapnik에 의하여 제안된 SVM(Support Vector Machine)은 고차원의 분류 문제에 탁월한 분류 능력을 발휘하고 있다[1].

전용 하드웨어로 구현하면 고차원의 데이터를 연산하기 위한 많은 연산을 병렬적으로 함으로써 연산시간을 줄일 수 있다. 최근에 SVM을 하드웨어화하여 실시간 처리가 요구되는 분야에 적용하려는 시도가 있는데, 크게 아날로그 구현과 디지털 구현 방법으로 진행되고 있다. 아날로그 SVM 하드웨어인 Kerneltron[2]는 내부적으로 아날로그 방식으로 설계하고 외부와는 디지털 신호로 변환하여 전달한다. 이 하드웨어는 빠른 커널 연산을 하여 물체 검출 등의 실시간 적용 분야에 초점을 맞추고 있지만 칩 안에서 학습까지 이루어지지 않아 재현 동작만 온라인으로 칩 상에서 이루어진다. 디지털 SVM 하드웨어인 디지털 SVM[3, 4]은 하드웨어에 적합한 학습 알고리즘을 제안하고 학습까지 이루어지는 디지털 SVM 하드웨어를 구현하였다. 하지만 이 디지털 하드웨어는 커널 연산 부분은 칩 상에서 동작하지 않음으로 인하여 시간 소모가 많은 커널 연산 과정에 의하여 처리 시간상 병목 현상이 발생하게 되며 또한 칩만의 독자적 연산 수행(stand-alone operation)이 불가능하다는 단점이 지적된다.

이 논문에서 제안하는 CSVM(Concurrent Support Vector Machine)은 계산속도가 빠른 커널 계산부를 내장형으로 만들어서 한 하드웨어에서 SVM의 모든 과정이 수행되는 통합기능을 구현하였다. 기존에 사용되는 SVM 학습 알고리즘인 QP(Quadratic Programming) 방법은 연산이 복잡하고 많은 메모리를 소모하여 하드웨어 구현에 부적합하다. CSVM은 반복 갱신 방식으로 하드웨어 구현이 용이하고 수렴성이 보장되는 SVM 학습 알고리즘의 하나인 KA(Kernel Adatron)[5] 및 KP(Kernel Perceptron)[4] 알고리즘을 사용한다. 또한 많은 연산이 요구되는 커널 연산을 병렬적으로 수행함으로써 소요시간을 단축시키고자 하며, 커널 값을 일정한 범위에 제한하는 커널 스케일링 방법[6]을 사용하여 하드웨어의 면적을 최소화한다. 고차원의 데이터를 가지는 암 진단 분류 문제 등에 적용하며 온라인상에서 즉시(point-of-care) 진단 결과를 요구하는 분야에서 적합한 구조를 지닌다.

2. Support Vector Machine

SVM은 기존의 신경회로망에서 이용된 경험적 위험을 최소화(empirical risk minimization)하는 원리보다는 구조적 위험을 최소화(structural risk minimization)하는 근사적 방법이고 통계적 학습 이론에 기반하여 최적 분류를 함으로서

* 正 會 員 : 仁荷大 工大 電氣工學科 博士課程
** 正 會 員 : 仁荷大 工大 情報通信工學部 敎授 · 工博
接受日字 : 2004年 5月 3日
最終完了 : 2004年 6月 21日

뛰어난 일반화(generalization) 성능을 보여 준다[7]. SVM은 커널 함수에 의하여 정의된 특징 공간(feature space)에서 초평면(hyperplane)을 경계로 학습 데이터를 분리한다. 학습 데이터 중에서 초평면과 가장 근접한 데이터들의 거리의 합인 마진이 최대가 되는 다차원 평면을 찾는다. 통계 학습 이론에 의하면 초평면의 일반화 성능은 속해 있는 공간의 차원이 아닌 마진에 의하여 결정되므로 고차원에서도 분류 능력이 뛰어나다.

학습 데이터가 $(\boldsymbol{x}_1, y_1),\cdots,(\boldsymbol{x}_n, y_n)$이고 입력패턴 $\boldsymbol{x}_i \in \mathcal{R}^d$ (d: 입력 공간의 차원)에 대하여 두 개의 클래스 $y_i = \{+1, -1\}$로 분류되는 문제를 생각해 보면, 그림 1과 같이 결정 함수인 $f(\boldsymbol{x})=\text{sign}((\boldsymbol{w}\cdot\boldsymbol{x})+b)$의 정규(normal) 벡터 $\boldsymbol{w}$와 바이어스 b는 마진을 최대화하도록 결정된다.

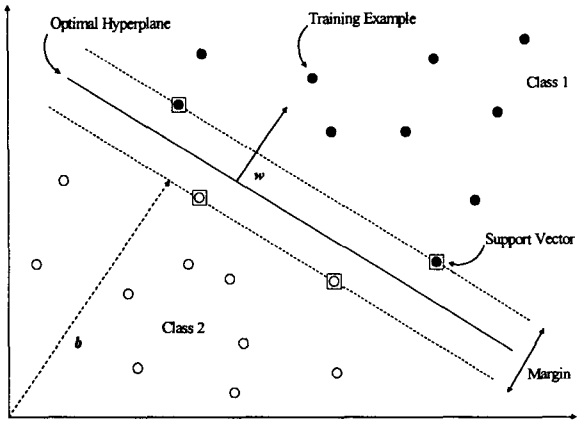

그림 1 선형 최대 마진 분류기
Fig. 1 A linear maximal margin classifier

SVM은 데이터를 비선형 커널 함수를 이용하여 다른 내적 공간, 즉 특징 공간으로 사상하여 비선형적인 초평면을 만들어서 비선형 분류 문제를 해결한다. 비선형 커널 함수에 의하여 SVM은 식 (1)과 같은 비선형 함수를 학습하게 된다.

$$f(\boldsymbol{x}) = sign\left(\sum_{i=1}^{n}\alpha_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b\right) \tag{1}$$

식 (1)에서 파라미터인 α는 선형적으로 제한적인 QP 문제에 의하여 결정된다. 식 (3)의 제한적 조건 하에서 식 (2)를 최대화하는 α를 구하게 된다.

$$Q(\alpha) = \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{i=1}^{n}\alpha_i\alpha_j y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) \tag{2}$$

$$\sum_{i=1}^{n}\alpha_i y_i = 0, \quad 0 \le \alpha_i \le C, \quad i = 1,\cdots,n \tag{3}$$

식 (3)에서 C는 경험적인 오차와 복잡도 간에 교환(trade-off)을 결정하는 조정 상수이다. s개의 support vector들이 존재한다면 b 값은 식 (4)에 의하여 구할 수 있다.

$$b = \frac{1}{s}\sum_{j=1}^{s}\left(y_j - \sum_{i=1}^{n}\alpha_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}_j)\right) \tag{4}$$

위와 같은 방법으로 SVM의 파라미터를 구하는 QP 방법은 계산량이 많고 구현하기 어렵다. 본 논문에서는 QP 방법이 아닌 구현하기 쉬운 KA와 KP 학습 알고리즘을 사용한다.

그림 2는 SVM의 구조를 나타낸다. 입력 $\boldsymbol{x}$와 support vector $\boldsymbol{x}_i$는 커널 함수에 의하여 비선형으로 사상된다. 최종적으로 계산된 $f(\boldsymbol{x})$ 값의 부호에 따라서 입력 $\boldsymbol{x}$의 클래스를 결정하게 된다.

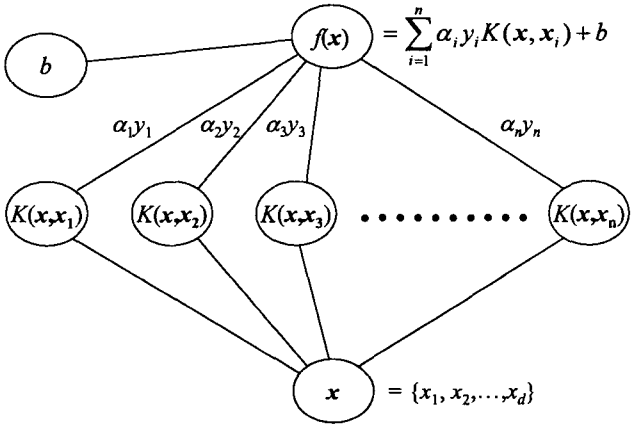

그림 2 SVM의 기본 구조
Fig. 2 Fundamental architecture of the SVM

표 1 KA 알고리즘
Table 1 KA algorithm

{초기화}
모든 i에 대해서 $\alpha_i = 0$, 학습율 η을 세팅
{학습 루프}
repeat
  for $i = 1,\cdots,n$
    $z_i = \sum_{j=1}^{n}\alpha_j y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j)$, $\Delta\alpha_i = \eta(1 - y_i z_i)$
    if $(\alpha_i + \Delta\alpha_i) > 0$ **then** $\alpha_i \leftarrow \alpha_i + \Delta\alpha_i$
    else $\alpha_i \leftarrow 0$
  end for
{종료 조건}
until (최대학습회수에 도달하거나
  $r = 0.5\times[\min_{\{i|y_i=+1\}}(z_i) - \max_{\{i|y_i=-1\}}(z_i)] \fallingdotseq 1$)

2.1. Kernel Adatron

KA 알고리즘은 통계적인 학습 방법의 하나인 adatron 알고리즘에 커널을 사용함으로서 비선형 feature 공간에서 마진을 최대화하게 한 것이다. Adatron 알고리즘은 이론적으로 최적해로의 수렴이 알고리즘의 반복에 지수 함수적으로 빠르게 일어난다고 알려져 있다[5]. 결국 KA 알고리즘은 SVM의 특징공간에 adatron을 도입하는 것으로 용이한 구현성 뿐만 아니라 커널에 의한 비선형 특징공간에서의 단순한 동작 특성도 동시에 얻을 수 있어, 기존 QP 학습 알고리즘이 가지는 계산과 구현상의 제약들을 효과적으로 해결할 수 있다. 본 논문에서는 표 1과 같은 바이어스를 사용하지 않는 KA 알고리즘이 사용된다.

2.2. Kernel Perceptron

KP 알고리즘은 퍼셉트론의 $\boldsymbol{w}$ 영역으로부터 SVM의 α영역으로 바꾼 것이다[4]. KP 알고리즘은 Rosenblatt의 퍼셉트론 알고리즘과 마찬가지로 수렴성이 보장된다. KP 알고리즘을 정리하여 표 2에 나타내었다.

표 2 KP 알고리즘
Table 2 KP algorithm

{초기화}
모든 i에 대해서 $\alpha_i = 0$, $b = 0$으로 세팅
{학습 루프}
repeat
  for $i = 1,\cdots,n$

$$o_i = \sum_{j=1}^{n}\alpha_i q_j + b, \quad q_{ij} = y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

  if $o_i \le 0$ **then** update $\alpha_i \leftarrow \alpha_i + 1$,
  $b = b + y_i$
  end for
{종료 조건}
until (최대학습회수에 도달하거나 모든 i 에 대하여 $o_i > 0$)

3. CSVM 구조

3.1. 주요 블록과 신호

CSVM의 구조는 그림 3과 같다. 본 하드웨어는 SVM의 마진 상에 위치하는 support vector를 기본단위로 하여 SVE(Support Vector Element) 모듈을 병렬 처리하여 연산속도를 증가시켰다. SVE의 개수는 학습샘플의 개수이며 학습결과 0이 아닌 α_i 값을 갖는 SVE가 support vector가 된다. 학습샘플의 개수가 많을 경우는 칩을 확장하여 사용 가능하게 설계되었다. 모든 SVE 간은 공유버스로 연결을 시켜서 커널 연산이 동시에 이루어지게 하였다. KA와 KP 학습 알고리즘의 유사한 점을 이용하여 칩에 KA, KP 두 가지 학습방법을 함께 내장하였고 SVE를 이용하여 커널 계산, 학습과 재현 동작을 모두 함으로써 적은 면적에 효과적으로 설계하였다. LE는 학습 단계의 파라미터 갱신을 위한 계산 시에만 사용된다.

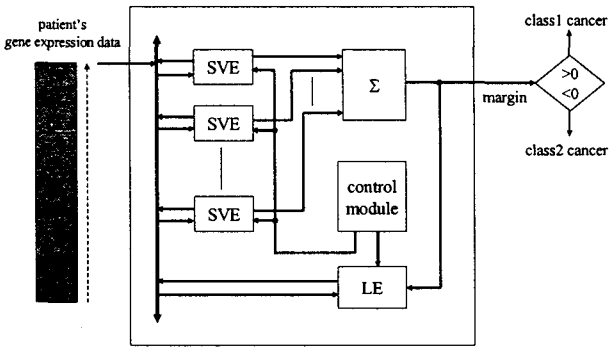

그림 3 질병진단을 위한 CSVM 구조
Fig. 3 CSVM architecture for cancer classification

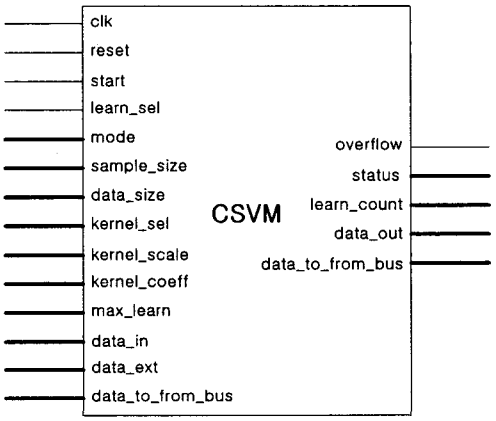

그림 4 CSVM의 입·출력 신호
Fig. 4 Input and output signals of CSVM

CSVM의 입력과 출력 신호는 그림 4와 같다. 데이터의 시작을 칩이 인식하도록 칩으로 data_in 신호를 최초로 넣을 때 start신호를 1로 해 준다. 샘플이 상당히 많은 데이터를 분류할 때 m개의 CSVM 칩을 함께 사용하기 위하여 i-1번째 CSVM의 data_out 출력 신호를 i번째 CSVM의 data_ext 핀을 통해서 입력을 받고 두 개의 CSVM의 공유 데이터 버스 입출력핀인 data_to_from_bus를 서로 연결해 준다.

mode 입력 신호는 칩의 동작을 결정한다. mode=0에서는 학습 동작만 이루어지고 mode=3에서는 학습이 종료된 후 재현 동작도 연속으로 이루어진다. 재현 동작만을 할 경우, 칩 내에 이미 저장되어 있는 파라미터를 이용할 때는 mode=1을 설정하면 되고 외부에서 파라미터를 입력 받아 사용하고자 할 때는 mode=2로 설정한다.

출력 신호 status 신호는 현재 칩의 상태를 사용자에게 알려 준다. status=0은 대기상태, status=1은 학습 동작 상태

를 나타내며, status=2는 재현 동작 상태, status=3은 동작 종료 상태를 알려주게 된다.

칩의 동작은 로딩, 커널 계산, 학습, 재현 단계로 나누어진다.

1) 로딩 단계: 외부로부터 $\boldsymbol{x}$, y를 로드하는 단계이다. 한 샘플에 해당하는 $\boldsymbol{x}$, y는 하나의 SVM의 내부 메모리로 저장되고 이 동작은 병렬이 아닌 순차적으로 이루어진다.

2) 커널 계산 단계: SVE_i에서 $\boldsymbol{x}_i$를 공유버스로 보내면 모든 나머지 SVE_j $(j \neq i)$들이 이를 동시에 받아 자신의 $\boldsymbol{x}_j$ 와 커널 함수 $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$를 계산한다. 이 동작은 병렬적으로 수행됨에 따라 순차적인 방법보다 속도가 n배만큼 증가된다.

3) 학습 단계: KA 또는 KP 학습 알고리즘을 이용하여 마진을 최대화하는 α_i 와 b를 찾는 과정이다. SVE와 LE가 모두 사용되며 LE에서 계산된 새로운 α_i 와 b는 공유버스를 통하여 해당 SVE의 메모리로 보내져서 종료조건이 만족될 때까지 학습 과정을 반복하게 된다.

4) 재현 단계: 테스트 샘플에 대해서 마진 값은 식 (1)에 의하여 계산된다. 마진 값이 양수이면 1계열 암으로, 음수이면 2계열 암으로 판정한다. 재현 단계에서도 병렬 동작으로 순차적 동작보다 n배 빠른 연산을 하게 된다.

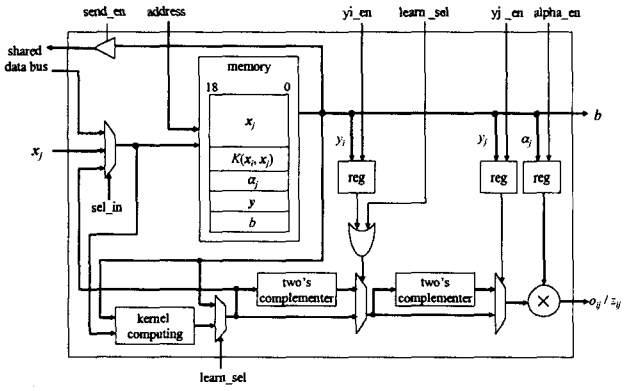

그림 5 Support Vector Element
Fig. 5 Support Vector Element

3.2. 각 블록의 구조

3.2.1. Support Vector Element

SVE는 그림 5와 같이 공유버스 인터페이스부와 메모리부, 식 (5)의 연산을 하는 부분으로 이루어진다. SVE의 출력 신호인 z_{ij}와 o_{ij}는 식 (5)에 의하여 계산된다.

$$z_{ij} = \alpha_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}_j) \text{ 또는 } o_{ij} = \alpha_i y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) \quad (5)$$

z_{ij}는 KA학습 또는 재현 동작에 사용되고 o_{ij}는 KP학습 동작에서만 사용되므로 KA학습 또는 재현 동작에서는 y_i 곱셈 연산은 이루어지지 않는다. y_i 값은 1 또는 -1 값이기 때문에 곱셈기 대신에 보수 변환기(two's complementer)를 사용하여 면적을 절약하였다. 재현 단계에서는 커널 계산부(kernel computing element)에서 신호를 받아서 y_i를 곱하게 되고 학습 단계에서는 메모리에서 이미 계산되어 저장된 커널 값을 받아서 y_i 값을 곱하게 된다.

메모리에 저장되어있는 $\boldsymbol{x}_j$와 y는 로딩 단계에서 저장되고 커널 연산 단계에서 $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$가 계산되어 메모리의 해당부분에 저장된다. α_i 와 b는 매회 학습마다 갱신되어서 메모리에 저장된 후 각 해당 레지스터에 저장된다. 하나의 메모리 출력 포트로부터 여러 파라미터를 레지스터로 보내야 하기 때문에 각각의 레지스터들은 해당 파라미터가 메모리로부터 도착할 때만 해당 yi_en, yj_en, alpha_en 신호를 1로 하여 저장한다. 원하지 않는 신호가 도착했을 때 저장하지 않고, 이전 값을 유지하기 위하여 해당 yi_en, yj_en, alpha_en 신호를 0으로 설정한다.

SVE의 입력은 공유버스 값, 외부 입력 값, 메모리에 저장될 커널 연산 값 중에 선택된다. 데이터를 보내고자 하는 SVE는 send_en=1로 하여 신호를 보내게 되며 한꺼번에 두 개의 SVE에서 신호를 보낼 수 없다.

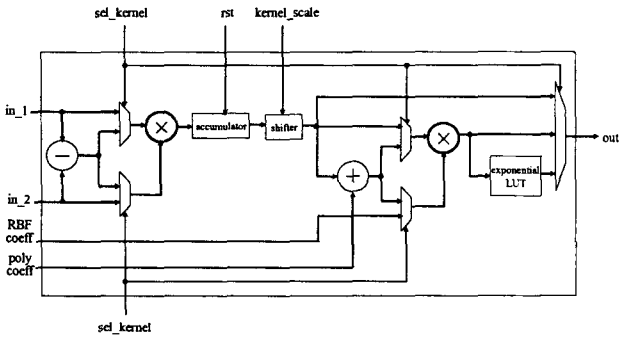

그림 6 커널 계산부
Fig. 6 Kernel computing part

표 3 커널 함수
Table 3 Kernel functions

SVM 타입	커널 함수
선형 SVM	$\boldsymbol{x}_i^T \boldsymbol{x}_j$
다항 SVM	$(\boldsymbol{x}_i^T \boldsymbol{x}_j + p)^2$
RBF SVM	$\exp(-c \parallel \boldsymbol{x}_i - \boldsymbol{x}_j \parallel^2)$, where $c = 1/2\sigma^2$

3.2.2. 커널 계산부

커널 함수는 linear, polynomial, RBF 커널을 그림 6과 같이 내장하였고 각 함수의 식은 표 3과 같다.

커널 함수 모듈은 면적을 최소화하기 위하여 곱셈기 2개만으로 3종류의 커널 함수를 설계하였다. 첫 번째 곱셈기에서는 선형, 다항 함수의 $\boldsymbol{x} \cdot \boldsymbol{x}_i$ 연산을 하고 RBF 함수의

$(x-x_i)\cdot(x-x_i)$ 연산을 한다. 두 번째 곱셈기에서는 다항 함수의 $(x\cdot x_i)\cdot(x\cdot x_i)$ 연산을 하고 RBF함수의 $c\cdot\|x_i-x_j\|^2$ 연산을 한다. RBF 연산에서 두 번째 곱셈기의 출력 값은 지수함수 LUT로 통과하게 된다. LUT의 입력비트의 크기에 비례하여 메모리 공간을 차지하므로 실험을 통하여 입력비트의 크기를 결정하였다.

고차원의 데이터의 커널 계산 시 커널 값이 너무 커질 수 있으므로 커널 스케일링 방법을 사용하여 커널 계산 값의 범위를 줄여서 데이터 비트의 정수부분을 효과적으로 절약시켰다.

3.2.3. 학습부

학습부는 파라미터 α_i, b 를 학습 종료 조건이 만족할 때까지 갱신하는 부분으로서 KA와 KP 알고리즘에 따라 그림 7과 같이 구현된다. 학습부는 커널 값을 사용하여 모든 SVE에서 계산되어서 더해진 o_i 또는 z_i 을 받고 갱신되어야 할 SVE$_i$의 메모리에 저장되어 있는 파라미터인 α_i^{old}, b^{old}, y_i 값을 공유버스를 통하여 받는다. KALE(Kernel Adatron Learning Element)와 KPLE(Kernel Perceptron Learning Element)에서 계산되어진 갱신된 α_i^{new}, b^{new} 값은 send_en 신호가 활성화되면 공유버스를 통하여 SVE$_i$의 메모리에 저장된다. 공유버스로부터 온 값들을 load신호에 따라 α_i^{old}, b^{old}, y_i 레지스터에 차례대로 저장한다. 학습 종료 조건이 만족하면 학습 단계는 끝나게 되고 최종 파라미터 α_i 와 b가 모든 SVE의 메모리에 저장되게 된다.

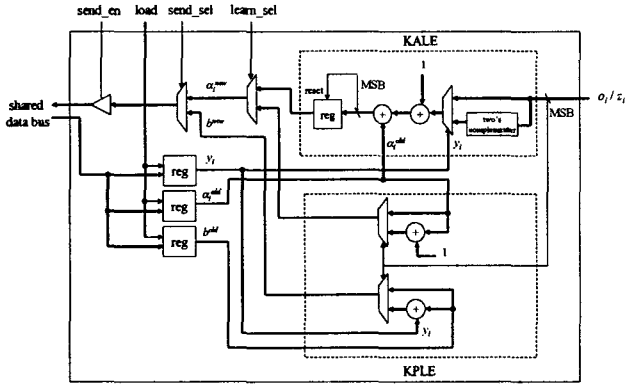

그림 7 학습부

Fig. 7 Learning Element

4. 실험결과

우리의 하드웨어의 성능을 입증하기 위하여 바이오인포매틱스 문제와 수중 음파(sonar)분류문제에 적용하여 연산시간과 성능을 평가한다. 바이오인포매틱스 문제로서 DNA chip으로부터의 고차원의 유전자 발현 데이터를 이용하여 백혈병 암의 종류를 분류하고자 한다. 또한, 분류기의 성능을 비교하는 데 많이 사용되는 벤치마크 데이터인 수중 음파 데이터로 실험을 하였다. 수중 음파 데이터의 분류 문제는 데이터의 차원이 높고 학습데이터와 테스트데이터 간의 중복 정도가 낮아서 어려운 문제로 알려져 있다.

4.1. 부동소수점과 고정소수점 실험

고정소수점 연산을 수행하는 하드웨어를 사용하여 알고리즘을 구현할 때 제한된 데이터 폭으로 연산을 하므로 양자화오차가 발생한다. 이 양자화오차에도 불구하고 시스템의 성능을 저하시키지 않고 부동소수점 연산과 동일한 성능을 얻을 수 있는 최소의 데이터 폭을 찾는 실험을 수행하였다.

표 4 다양한 데이터 폭과 LUT의 어드레스 비트 수에 대한 수중 음파 테스트 데이터 오분류율

Table 4 Percentage of misclassified test patterns on sonar dataset

데이터 폭	LUT의 어드레스 비트 수			
	12	14	16	18
20	14.4%	19.2%	15.4%	15.4%
22	15.4%	6.7%	6.7%	6.7%
24	9.6%	6.7%	6.7%	6.7%

표 5 다양한 데이터 폭과 커널 스케일율에 대한 백혈병 데이터의 테스트 오차 개수

Table 5 Test errors on leukemia dataset for different data widths and kernel scales

데이터 폭	커널 스케일율				
	1	4	16	64	256
14	14	15	17	13	1
16	20	18	17	1	1
18	20	17	1	1	1
20	16	1	1	1	1

60개의 특징(feature)을 가진 수중 음파 데이터는 104개의 학습패턴과 104개의 테스트 패턴으로 되어 있다. KP 학습과 RBF커널 함수를 사용한 부동소수점 실험 결과 104개의 테스트 패턴 중에 7개의 패턴만을 오분류하였다. 이는 Anguita et al.[5]의 연구결과보다 우수한 결과이다. KP 알고리즘이 9개의 패턴을 오분류한 KA 알고리즘보다 성능이 좋았다. 표 4는 다양한 데이터 폭과 LUT의 어드레스 비트 수에 대한 고정소수점 실험 결과이다. 데이터의 정수 부분은 6비트로 고정하였다. 22비트의 데이터 폭과 14비트의 LUT의 어드레스 비트 수 이상일 때 부동소수점 알고리즘과 성능이 동일했다.

백혈병 데이터[8]는 DNA 마이크로어레이칩으로부터 얻은 유전자 발현 양상 데이터이다. 이 데이터는 72명의 환자에 대한 각 7129개의 유전자로 이루어져 있다. 72명의 환자 샘플 중에 38명의 데이터는 학습을 위해서 사용되었고 나머지 34명의 데이터는 테스트용으로 사용되었다. 전처리 과정[9]을 통해서 얻은 3571개의 유전자를 실험에 사용하였다. KP 알고리즘과 RBF커널 함수를 사용한 부동소수점 알고리즘 실험을 통해서 단지 1개의 테스트 패턴을 오분류하였다. 표 5는 고정소수점 알고리즘을 사용하여 다양한 데이터 폭과 커널 스케일율에 따라 얻은 테스트오차의 수를 나타내고 있다. 데이터의 소수 부분은 12비트로 고정하였다. 데이터 폭이 14비트로 감소할지라도 256의 비율로 커널 스케일을 한다면 테스트 오차는 1로 감소한다. 데이터의 차원이 커진다면 커널 스케일율을 크게 정해주어야 좋은 결과를 얻을 수 있다. 비교적 적은 60개의 특징을 가진 수중 음파 데이터에서는 커널 스케일 방법의 효과가 덜 했다.

표 6 HDL 합성 결과
Table 6 HDL synthesis results

블록	Registers	Counters	Mux	Tristates	Shifters	Add/Sub	Comparators	Multipliers	BRAM
커널 계산부	2	-	8	-	1	2	2	3	-
SVE	8	-	11	1	1	4	2	3	3
LE	12	-	6	1	-	6	2	-	-
제어부	28	3	-	-	12	24	26	-	-
CSVM	367	3	446	41	52	231	108	120	120

표 7 각 단계에 소요되는 사이클 수
Table 7 The number of cycles needed for each phase

단계	로딩	커널 계산	학습 1회	재현
사이클 수	135,779	136,154	18	3,586

4.2. FPGA 구현

CSVM은 Verilog HDL로 설계되었으며 Xilinx Virtex2 FPGA XC2V8000에 구현하였다. XC2V8000 FPGA는 8백만 게이트들을 구현 가능하고 168개의 18비트 전용 곱셈기와 168개의 18,000비트 독립적인 메모리 블록(BRAM)을 사용할 수 있다. CSVM 칩은 25MHz로 동작하고 재현단계에서 1.67×106 data/s의 throughput을 나타냈다.

표 6은 각 블록과 CSVM을 합성한 결과 소요되는 연산소자들의 개수를 나타내었다. CSVM은 백혈병 데이터 문제에 적합한 크기인 40개의 SVE와 4000개의 데이터를 처리 가능하도록 설계하였다. 설계 결과 각 단계에 필요한 사이클은 표 7과 같았다. 백혈병 데이터를 KP 알고리즘을 사용하여 모든 단계에 걸리는 시간은 단지 11ms에 불과했다.

5. 결 론

우리는 공유 데이터 버스를 이용한 병렬 연산과 온칩 커널 연산을 사용하는 효율적인 CSVM 하드웨어 구조를 제안하였다. 병렬 구조와 공유 버스를 이용한 concurrent 동작으로 빠른 SVM 연산이 가능하고 특히 고차원의 데이터의 분류 문제에 적합하다. 많은 계산을 요구하여 지금까지 호스트컴퓨터에서 off-line으로 수행하던 SVM의 커널 연산을 하드웨어 위에서 공유 데이터 버스를 이용하여 빠르게 처리하였다. 커널 연산을 포함한 SVM에서 필요한 모든 연산이 CSVM 하드웨어 위에서 한 번에 빠르게 이루어지므로 온라인, 독립적인 수행(stand-alone)이 요구되는 적용분야에 사용 가능하다. 커널 스케일링 방법은 많은 데이터의 특징을 요구하는 바이오인포매틱스 분야의 패턴분류문제에 효율적이다.

기본 엘리먼트 SVE가 많은 면적을 소모하는 메모리와 곱셈기를 포함하여 많은 샘플 데이터를 가진 대상에 적용한다면 면적이 너무 커지므로 비용이 오르는 단점이 있다. 또한 SVE의 개수가 많아질수록 병렬성이 증대되어 속도가 더 증대되므로 한정된 하드웨어 공간에 더 많은 SVE를 포함시켜야 한다. 많은 종류의 실시간 적용 분야에 CSVM을 적용하고 병렬성을 극대화하기 위하여 SVE의 면적을 최소화하는 연구가 필요하다.

감사의 글

본 연구는 2004년도 인하대학교의 지원 (INHA-31544) 에 의하여 이루어진 연구로서, 관계부처에 감사드립니다.

참 고 문 헌

[1] Furey T.S., Cristianini N., Duffy N., Bednarski D.W., Schummer M. and Haussler D., "Support vector machine classification and validation of cancer tissue samples using microarray expression data", Bioinformatics, Vol. 16, No. 10, pp. 906-914, 2000.

[2] Genov, R. and Cauwenberghs, G., "Kerneltron: support vector "machine" in silicon", IEEE Transactions on Neural Networks, Vol. 14, No. 5, pp. 1426-1434, 2003.

[3] D. Anguita, A. Boni and S. Ridella, "A digital architecture for support vector machines: theory,

algorithm and FPGA implementation", IEEE Transactions on Neural Networks, Vol. 14, No. 5, pp. 993-1009, 2003.

[4] D. Anguita, A. Boni and S. Ridella, "Digital kernel perceptron", Electronics Letters, Vol. 38, No. 10, pp. 445-446, 2002.

[5] T.T. Friess, N. Cristianini and C. Campbell, "The kernel-adatron algorithm: a fast and simple learning procedure for support vector machines", 15th International Conference on Machine Learning, Wisconsin, USA, pp. 188-196, July 1998.

[6] B. Scholkopf and A. J. Smola, Learning with kernels: support vector machines, regularization, optimization, and beyond, MIT Press, 2002.

[7] Vapnik V.N., Statistical Learning Theory, John Wiley and Sons, New York, 1998.

[8] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander, "Molecular classification of cancer: class discovery and class prediction by gene expression monitoring", Science, Vol. 286, pp. 531-537, 1999.

[9] S. Dudoit, J. Fridlyand and T. P. Speed, "Comparison of discrimination methods for the classification of tumors using gene expression data", Journal of the American Statistical Association, Vol. 97, No. 457, pp. 77-87, 2002.

[10] Nello C. and John S. T., An Introduction to Support Vector Machine, Cambridge University Press, 2000.

저 자 소 개

위 재 우 (魏 載 玗)

1974년 5월 21일생. 1997년 인하대 전기공학과 졸업. 1999년 동 대학원 전기공학과 졸업(공석). 1999년-현재 동 대학원 전기공학과 박사과정

Tel : 032-860-7396, Fax : 032-863-5822

E-mail : wizet2000@empal.com

이 종 호 (李 鍾 浩)

1953년 4월 14일생. 1976년 서울대 전기공학과 졸업. 1978년 동 대학원 전기공학과 졸업(공석), 1986년 미국 아이오와주립대 전기 및 컴퓨터 공학과 졸업(공박), 1979년-1982년 해군사관학교 전임강사, 1980년-1982년 국방과학연구소 위촉연구원, 1986년-1989년 미국 노틀담대학교 조교수, 1991년-1993년 대한전기학회 컴퓨터 및 인공지능연구회 간사장, 1994년-1995년 미국 브라운대학교 방문교수, 1997년-1998년 인하대 집적회로설계센터 소장, 1989년-현재 인하대학교 정보통신공학부 교수, 2000년-현재 수퍼지능기술연구소 소장

Tel : 032-860-7396, Fax : 032-863-5822

E-mail : chlee@inha.ac.kr