CMAC 신경회로망을 이용한 패턴인식 학습의 개선

°김종만, 김성중* , 권오신** 김형석***
*전북대학교 제어계측공학과, **군산대학교 제어계측공학과, ***국방과학연구소

The Improvement of Pattern Recognition using CMAC Neural Networks

Jongman Kim, Sungjoong Kim*, Ohsin Kwon**, Hyongsuk Kim***
*Chonbuk National Uni. **Kunsan National Uni. ***Agency for Defence Develop.

Abstract

CMAC (Cerebeller Model Articulation Controller) is kind of Neural Networks that imitate the human cerebellum. For storage and retrieval of learned data, the input of CMAC is used as a key to determine the memory location. he learned information is distributively stored in physical memory. The learning of CMAC is very fast and converged well, therefore, it effects the application of Pattern Recognition. Through the our experiment of Pattern Recognition, we will prove that CMAC is very suitable for On-line real time processing and incremental learning of Neural Networks.

1. 서 론

신경회로망(Neural Network)은 현대 제어나 패턴인식 등 많은 분야에 응용되고 있으며, 특히 패턴인식(Pattern Recognition) 부분에 아주 성공적으로 응용되고 있다. [1]

그것은 신경회로망으로 구성된 패턴인식 시스템이 기존의 패턴인식 시스템에 비하여 일반적으로 다음과 같은 장점을 갖고 있기 때문이다.[2]

첫째, 패턴인식 문제는 구조적으로 병렬성을 내재하고 있기 때문에 신경회로망의 병렬처리 능력을 이용하여 신속한 처리가 가능하다. 둘째, 입력 패턴이 미리 훈련된 패턴에 비해 약간의 노이즈가 있는 형태일지라도 잘 인식할 수 있다. 셋째, 훈련(training)이 되지 않은 새로운 입력 패턴에 대해서도 가장 비슷한 부류의 패턴을 찾아내고, 넷째, 구조가 간단하며 프로그램 과정에서도 복잡한 소프트웨어 코딩의 필요시 단지 노드간의 연결 링크에 대한 연결 강도(weight)만을 학습(learning)시키면 된다.

패턴인식과 제어 등에 응용되는 신경회로망의 모델은 역전파(Backpropagation) 알고리즘, Hopfield 모델, Grossberg Networks, CMAC 신경회로망등이다. 이 중 대부분이 역전파 알고리즘을 가장 많이 사용하고 있는데 이 알고리즘은 수렴하는데 많은 시간이 걸리므로 On-line 실시간 학습에 적합하지 않으며, 학습하는 동안 많은 양의 계산을 필요로 한다.

또한 local minima에 빠질 우려가 있고 추가 학습을 할 수 없다. 반면 1975년에 처음 Albus에 의해 제안되었고, 계속 발전되어온 CMAC(Cerebellar Model Articulation Controller) 신경회로망은 매우 넓은 범위의 함수로부터 비선형 관계를 학습할 수 있고, 학습이 매우 빠르며, 수렴 특성이 좋으므로 신경회로망 응용에서 좋은 알고리즘이라 할 수 있다. 그러나 기존의 CMAC은 함수 및 제어분야에서 학습용으로만 이용되어 왔다.[1-6] 이 CMAC의 특징들을 패턴인식에 이용할 경우 On-line 실시간 처리와 추가 학습문제등 기존의 패턴인식의 어려운 문제를 해결할 수 있다. 따라서 본 논문에서는 CMAC을 이용한 패턴인식을 제안하고, 문자인식 실험을 통하여 그 가능성을 입증하여 보이고자 한다.

2. CMAC 신경회로망

2.1 CMAC의 구성

* CMAC mapping

CMAC은 1975년 Albus가 발표한 소뇌(cerebellum)의 모델로써 근본적으로 feedforward network 또는 table-look-up 모들이지만 다른 신경회로망과는 다른 구조를 하고 있다. 다음 그림1은 소뇌의 기능을 인공적으로 모방한 CMAC 모델이다.

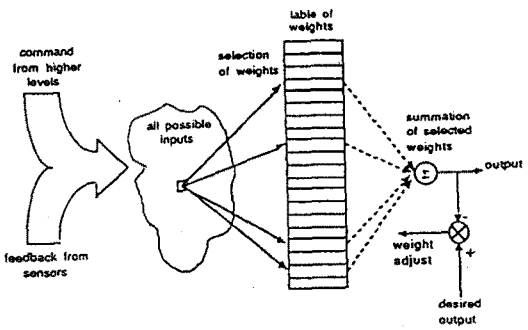


그림 1. CMAC 모델(Albus 1975)

CMAC은 다음과 같은 순차적 mapping으로 구성되어 있다.

S → M
M → A
A → P

여기서, S는 입력벡타(state input vector), M은 중간 변수(intermediate variables), A는 메모리 어드레스(physical memory address), P는 출력 벡타(output vector)이다.

CMAC에서 상태에 대한 정보가 메모리에 분산적으로 저장되고 address key로서 상태변수를 사용하여 검색한다.

데이타의 검색을 위하여 상태 변수가 몇개의 중간 변수로 mapping되고, 출력값 검색을 위해서 물리적인 메모리 어드레스로 정보가 저장된다.

첫번째 mapping(S→M)에 대하여, 각 상태변수 s는 그림 2의 단일 입력변수의 블럭분할에서 보여진 것처럼 몇 개의 블럭으로 나뉘어지고, 각 블럭은 다시 몇 개의 element로 나뉘어진다.

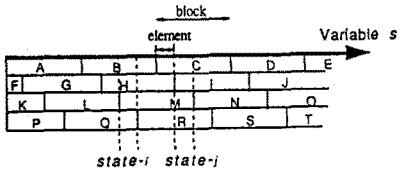


그림 2. 단일 입력 변수에 대한 CMAC의 블럭 분할

그림 2에서는 변수에 대하여 블럭이 네개의 분할 층으로 되어 있다. 단, 첫번째와 마지막 블럭은 불완전한 블럭으로 보통 블럭보다 더 적다. 각 층에서 각 블럭은 인식명(identification name)으로 배치된다. 그러므로 양자화된 상태는 각 층에 대한 블럭 이름들의 집합으로 특성화된다. 예를 들면, 그림 2에서 상태 i와 j는 각각 {B, H, M, Q}와 {C, I, M; R}에 의해서 특성화된다. Mapping은 다변수인 경우로 확장할 수 있는데, 그림 3은 변수가 두 개인 경우에 대한 예이다. 사각형 c1은 변수 s1의 B와 s2 의 b에 의해서 정의되고, c2는 H와 h에 의해서 정의된다. 고차원인 경우의 중간 변수들은 hypercube(초입방체)가 된다.

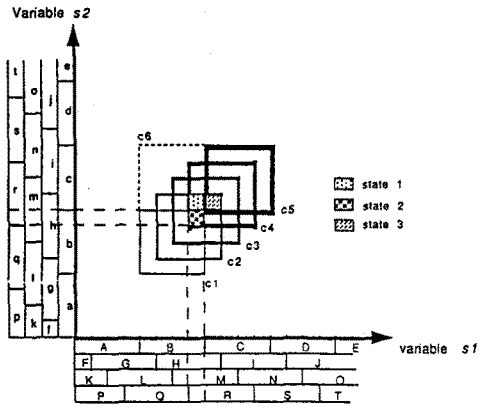


그림 3. 두개의 입력 변수에 대한 CMAC의 블럭 분할

두번째 mapping은 중간변수(intermediate variable)들이 물리적인 메모리 어드레스로 mapping 되는데(M→A), 이 mapping 과정에서 hash coding이 이용된다. Hash coding은 크고 간헐적으로 밀집된 큰 메모리 어드레스 공간을 보다 적고 밀접한 메모리 공간으로 압축하기 위한 메모리 어드레스 기법인데, 큰 공간속의 각각의 메모리 어드레스가 더 작은 공간의 어드레스로 mapping하여진다. mapping된 어드레스를 결정하는 방법은 0과 1 사이의 임의의 수를 발생시키도록 중간변수(각 hypercube의 ID 수)를 seed로 사용하고, 물리적 크기에 의한 임의의 수를 곱하는 방법이다. 따라서 hash coding을 사용함으로써 메모리를 효율적으로 이용할 수 있다.

마지막으로 상태에 대한 물리적 메모리 어드레스의 모든 내용은 출력을 만들기 위해 합해지는데 이 과정이 A → P mapping이다. 결과적으로 한 상태를 포함하는 물리적 메모리 영역의 모든 hypercube들은 상태의 출력을 계산하는데 공헌하게 된다. 각 hypercube는 여러 개의 상태를 포함하여 공유하게 된다. 즉, 그림 3에서 보는 바와 같이 상태 2는 hypercube c1, c2, c3 및 c4를 포함하고 상태 3은 c2, c3, c4 및 c5를 포함한다. 이와 같이 메모리를 공유하는 것은 CMAC이 좋은 일반화(generalization) 특성을 갖고 있음을 나타내며, Hypercube의 크기는 블럭 수에 의해서 결정되기 때문에 일반화의 정도가 조절될 수 있다.

CMAC 정보 검색을 수학적으로 기술하면 다음과 같다.

S(k)를 시간 k에서의 상태, Cj(S(k))를 상태 S(k)에 포함된 j 번째 hypercube, N_c 를 상태에 포함된 hypercube 의 수, ID_{CASIA} 를 Cj(S(k))의 ID(identification) 수라 하자. 벡터 $U(s(k))$ 는 다음과 같이 정의 한다.

$$U(s(k)) = [u_1(s(k)), u_2(s(k)), \ldots, U_j(s(k)), \ldots, U_{p_m}(s(k))]^T \quad (2.1)$$

$$u_i(s(k)) = \begin{cases} 1, \text{ if } i \in \{ADDR(ID_q(s(k))) ; 1 \le j \le N_c\} \\ 0, \text{ otherwise} \end{cases} \quad (2.2)$$

이고, P_m 은 물리적 메모리의 크기이다. 벡터 M은 물리적 메모리상에 있는 모든 내용을 다음과 같이 나타낸다.

$$M = [m1, m2, \ldots, m_{p_m}]^T \quad (2.3)$$

상태 S(k) 에 대하여 저장된 정보를 INFORM(S(k)) 라 하면 다음과 같이 계산된다.

$$\text{INFORM}(s(k)) = M \cdot U(s(k)). \quad (2.4)$$

한 개의 상태에 대한 정보는 그 상태에 할당된 여러개의 모델링 영역으로부터 검색되어진다

2.2 CMAC의 특성

앞에서 언급한 CMAC의 특성을 몇 가지로 요약하면 다음과 같다.

1) CMAC은 좋은 일반화(generalization)특성을 가지고 있다.

많은 메모리 cell들이 하나의 정보를 표현하기 위해 사용된다. 이것은 사람의 두뇌의 기능과 유사한데, key 특성으로 어드레스를 사용하여 메모리 cell의 연상(association)을 통하여 검색된다.

그러므로 CMAC은 불완전한 입력 패턴 key들을 사용하여 일반화 특성과 패턴 검색과 같은 연상 메모리의 특징을 가지고 있다.

2) CMAC은 좋은 학습 성능을 가지고 있다.

CMAC은 부분적으로 분포된 정보가 있는 신경회로망 중의 하나이다. 메모리에서의 입출력 관계는 이웃하는 영역 사이에서 결정된다. 이것이 학습을 쉽게 만들어준다.

3) CMAC은 추가 학습(incremental learning) 특성이 좋다.

훈련(training)을 위한 전체 데이타 집합을 미리 준비하는 것이 어려울 때 이 학습 데이타는 학습 과정에서 데이타에 추가시켜 새로운 데이타를 만든다. 따라서 weight를 가하여 데이타 집합의 새로운 확장 영역으로 갱신시킬 수 있다.

3. 패턴인식을 위한 CMAC NETWORK 구조

그림 4는 패턴인식을 위해서 제안한 CMAC 구조이다.

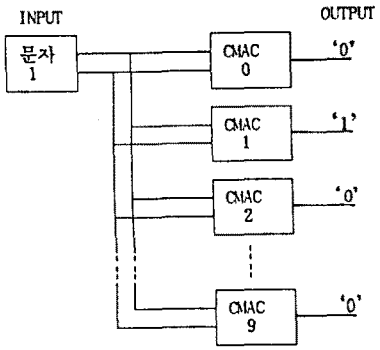


그림 4. 패턴인식을 위한 CMAC NETWORK 구조

먼저 본 문자인식을 위해서 숫자 0-9 까지의 입력 데이터를 미리 선정한다. 입력 데이타 (x,y)를 CMAC 에 인가했을 때 이에 대한 CMAC의 출력은 x축과 y축등 입력 변수를 통하여 형성된 모든 hypercube들의 고유 이름에 의해 지정되는 메모리 내용들의 값들로 얻어진다. 0-9 까지의 입력 문자중 "1"을 선정하여 CMAC NETWORK에 인가했을 때, 각 문자에 대한 출력 CMAC0, CMAC1,CMAC2, ... CMAC9 중 CMAC1에만 '1'에 근사한 값으로 얻어지고,나머지 CMAC은 '0'에 근사한 값으로 출력을 얻는다. 이와 같은 방법으로 나머지의 숫자에 대해서도 각각 수행하여 문자인식 실험을 수행한다.

4. 모의 실험 및 고찰

제안된 패턴 인식을 위한 CMAC의 학습은 0-9 까지의 모든 문자에 대하여 컴퓨터 모의 실험을 수행하였다. 이 실험에 사용된 학습 데이타는 각 숫자마다 100개씩 사용되었고, mapping 함수는 매 20회의 학습마다 갱신되었다.
그림 5는 사용된 각 문자에 대한 데이타를 도시화한 것이다.

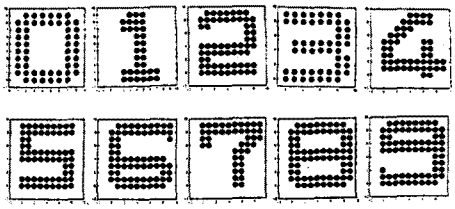


그림 5. 각 문자의 데이타 모양

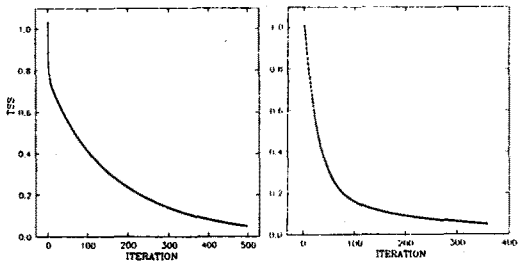
이들 데이타를 그림 4에서와 같은 방법으로 각각의 CMAC에 대하여 훈련(training)하여 학습된 결과 표 1과 같은 결과치를 얻었다.

표 1. 각 문자에 대한 CMAC 학습 Test 결과치

목표값	0	1	2	3	4
실험결과	0.830490 -0.011037 -0.000351 0.001887 -0.003192 -0.012835 0.000725 -0.006973 0.085293 0.077567	-0.048233 0.949765 -0.007691 -0.008493 -0.007340 -0.023555 -0.024051 -0.068499 0.004379 0.169535	0.033806 -0.056340 0.932816 -0.008890 -0.001518 -0.015066 -0.014232 -0.062204 -0.013973 0.166220	0.170547 -0.053688 -0.007510 0.931164 -0.005190 -0.008176 -0.017664 -0.063750 0.077551 -0.018064	-0.034874 -0.040979 0.000491 -0.003123 0.872942 0.003233 0.016075 -0.008895 0.145560 0.022741
목표값	5	6	7	8	9
실험결과	-0.112902 -0.028550 -0.003302 -0.001230 -0.001952 0.976648 -0.008556 -0.036778 -0.004295 0.160072	0.080936 -0.044840 -0.001896 -0.003966 0.007286 0.000826 0.895626 -0.030628 -0.081849 0.134522	-0.028355 -0.037895 -0.005275 -0.004583 -0.001420 -0.017119 -0.014335 0.935255 -0.063079 0.176603	0.114789 0.004903 0.000235 0.001983 0.004852 0.004631 -0.003423 -0.001355 0.832557 0.040599	0.077851 0.025877 0.003265 0.000095 0.000970 0.017245 0.011029 0.036845 0.061236 0.826559

이 도표에서 볼 때 각 숫자마다 훈련시킨 결과, 각 숫자에 대한 CMAC 만이 '1'에 가까운 값으로 출력되고, 나머지는 '0'에 가까운 값으로 학습 결과가 빠른 시간내에 산출되었음을 확인할 수 있었다.

이들 출력시킨 결과에 대한 TSS(Total Sum of Squared Error)를 각 iteration 마다 도시화한 것이 그림 6(a)(b)에 나타내주고 있다. 이때 각 문자에 대하여 5-10%정도의 노이즈를 가하여 수행시켜 봤을 때도 근사하게 얻어낼 수 있었다. 이들 문자중 '0'의 데이타에 대해 노이즈를 가하여 학습한 결과치를 표 2에 보였다.



(a) (b)

그림 6. CMAC에 의한 각 iteration별 패턴인식 학습 결과
(a) number 0일 때의 TSS (b) number 1일 때의 TSS

표 2. 임의의 수(0)에 대하여 NOISE(10%)를 가했을 때의 결과

TARGET VALUE (NUMBER 0)

output value t_sum=0.785252
output value t_sum=0.024749
output value t_sum=0.008391
output value t_sum=-0.000351
output value t_sum=0.001887
output value t_sum=0.010310
output value t_sum=-0.021963
output value t_sum=0.045697
output value t_sum=-0.007189
output value t_sum=-0.024207

5. 결 론

CMAC 신경회로망은 앞에서 언급된 바와 같이 학습 속도가 빠르며 추가 학습이 용이한다는 점등 여러가지의 특징을 가지고 있다.
본 논문에서 CMAC 신경회로망을 패턴인식에 적용하고, 각각의 문자인식 실험을 한 결과, CMAC 알고리즘이 패턴인식에 잘 구별되어 학습해 감을 확인할 수 있었다. CMAC이 학습속도면에서도 빠르고 또한, 각각의 데이타에 5-10% 정도의 노이즈를 가했을 때에도 CMAC 신경회로망을 통하여 빠른 속도내에 수렴해 가는 좋은 특성을 보였다.
따라서 CMAC 신경회로망은 On-line 실시간 처리에 적합하다는 것과 minima에 빠지지 않고 추가 학습을 통하여 패턴인식 응용에 좋은 결과를 입증할 수 있었다.

앞으로 CMAC 신경회로망에 대한 패턴인식 적용 문제중 글자의 크기 문제와 Rotation 문자에 대한 문제, 그리고 다양한 노이즈를 갖는 패턴에 대하여 더욱더 연구할 과제로 남아있다.

참 고 문 헌

1. Albus, J. S., "A new approach to manipulator control: The Cerebellar Model Articulation Controller (CMAC)," Journal of Dynamic Systems, Measurement, and Control, Transactions of the ASME, pp. 220-227, September 1975.
2. Albus, J.S., " Data storage in the Cerebellar Model Articulation Controller (CMAC)," Jounal of Dynamic Systems, Measurement, and Control, Transactions of the ASME, pp. 228-233, September 1975.
3. Barto, A. G., R.S. Sutton and P. Anandan, "Pattern-recognizing stochastic learing automata," IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC_15, no.3, pp. 360-375, May/June 1985.
4. Kim,H., CMAC Neural Network-based self-learing, Ph.D.dissertation, Dept. of Electical and Computer Engineering, U. of Missouri-Columbia, May 1992.
5. 한국과학기술원 산학 협동 공개 강좌, 신경 회로망 컴퓨터: 이론, 응용 및 구현, 1988.8, 1989.8, 1990.8.
6. Yoh-Han Pao, Adaptive Pattern Recognition and Neural Networks, Addison-Wesley Publishing Company, Inc., 1988.