알고리즘 수정에 의한 홉필드 모델의 성능 개선

°오 상훈, 윤 태훈, 김 재창
부산 대학교 전자공학과

Dummy Stored Memory Algorithm for Hopfield Model

Sang Hoon O, Tae Hoon Yoon, Jae Chang Kim
Pusan National University Dept. of Electronics Eng.

ABSTRACT Recently Hopfield proposed a model for content-addressable memory, which has been shown to be capable of storing information in a distributed fashion and determining the nearest-neighbor. Its application is, however, inherently limited to the case that the number of 1's in each stored vector is nearly the same as the number of 0's in that vector. If not the case, the model has high probability of failure in finding the nearest-neighbor. In this work, a modification of the Hopfield's model, which works well irrespective of the number of 1's (or 0's) in each stored vector, is suggested.

1. 서 론

광 정보 처리기술의 발달로 정보의 대량 병렬처리가 가능하게 되었으며, 이러한 장점을 이용한 뉴런망(neural networks)과 같은 분포 연상 기억(distributed associative memory)을 가진 광 컴퓨터에 대한 관심이 높아지고 있다. 이렇게 뉴런망을 광 기억 장치에 응용할려면 뉴런망에 대한 모델이 필요한데, 최근에 Hopfield가 뉴런망의 간단한 모델을 제시하였다.[1,2] 이 모델에서, 각각의 뉴런들이 시냅스(synapse)를 통하여 연결이 되어 있어서 외부에서 입력이 주어지면 뉴런 벡터는 기억되어 있는 뉴런 벡터들 중에서 입력과 가장 비슷한 벡터(nearest- neighbor vector)를 찾아간다. Psaltis 와 Farhat 는 Hopfield 모델의 광학적 구현 방법을 제시하였고,[3] Farhat 등은 홀로그래피를 이용하여 이 모델을 실현하였다.[4]

그런데, 이 모델은 자체적 결함이 존재하여서, 기억된 벡터 중에서 입력과 가장 비슷한 것이 항상 최후 안정 벡터가 되지 못하며, 더러는 잘못된 안정 벡터가 나타난다.[5] 이러한 현상이 나타나는 근본적인 이유로서 벡터내에 1과 0가 차지하고 있는 비트 수의 불균형을 들 수 있다.

여기서는 Hopfield 모델의 이러한 문제점에 대하여 알아보고, 이 문제점을 해결할 수 있는 방법을 제시하고자 한다.

2. Hopfield 모 델

Hopfield 모델은 시냅스를 통하여 상호 연결되어 있는 수 많은 뉴런들로 구성되어 있다. 이 모델에서, i번째 뉴런의 상태를 v_i라 하면, v_i는 0,1의 값을 가진다. 따라서, 뉴런들의 상태가 2진수 벡터로 표시되며, 이 뉴런 벡터들은 시냅스 행렬(synaptic matrix) T에 기억된다. 시냅스 행렬에 기억되어 있는 뉴런 벡터의 수를 M이라 하고, 한 뉴런 벡터의 비트 수를 N이라 하면, 시냅스 행렬에 기억된 뉴런 벡터들은 다음과 같이 표시된다.

$$V^{(m)} = (v_1^{(m)}, v_2^{(m)}, \cdots, v_N^{(m)}), \quad m = 1,2,\cdots,M \qquad (1)$$
$$v_i^{(m)} = 1 \text{ 또는 } 0$$

이 뉴런 벡터들이 기억되어 있는 시냅스 행렬 T의 요소 t_{ij}는 다음과 같다.

$$t_{ij} = \sum_{m=1}^{M} (2v_i^{(m)} - 1)(2v_j^{(m)} - 1) - M\delta[i-j] \qquad (2)$$

이 모델에서, $m = m'$인 2진수 벡터 $V^{(m')} = (v_1^{(m')}, v_2^{(m')}, \cdots, v_N^{(m')})$이 입력으로 주어지면, 입력과 시냅스 행렬간에 계산이 다음과 같이 행해진다.

$$\hat{v}_i^{(m')} = \sum_{j=1}^{N} t_{ij} v_j^{(m')} = \sum_{m=1}^{M} (2v_i^{(m)} - 1)\left[\sum_{j=1}^{N} v_j^{(m')}(2v_j^{(m)} - 1)\right] \qquad (3)$$

이때, 벡터(1)내에서 1,0의 수가 같으면, (3)식의 평균값은 $\frac{N}{2}(2v_i^{(m')} - 1)$이 된다. 즉, $v_i^{(m')} = 1$ 이면 $\langle \hat{v}_i^{(m')} \rangle > 0$ 이고, $v_i^{(m')} = 0$ 이면 $\langle \hat{v}_i^{(m')} \rangle < 0$ 이다. 그러므로, 뉴런의 상태 $v_i^{(m')}$의 시간적 진전이 다음과 같이 정의되면, 뉴런들은 시냅스 행렬에 기억된 $m = m'$인 벡터를 가지게 된다.

$$v_i^{(m')} = TH\{\hat{v}_i^{(m')}\} = \begin{cases} 1 & \hat{v}_i^{(m')} > 0 \\ 0 & \hat{v}_i^{(m')} < 0 \end{cases} \qquad (4)$$

임의의 입력 벡터 $V = (v_1, v_2, \cdots, v_N)$ 가 주어져서 이 알고리즘에 따라 뉴런들의 상태가 계속 변할때, 뉴런 벡터는 시냅스 행렬에 기억되어 있는 벡터 중에서 입력과 가장 비슷한 벡터로 시간적 진전이 진행되어 이것을 최후 안정

벡터로 가지게 된다.

3. Hopfield 모델의 문제점

2절에서 Hopfield 모델을 설명하였는데, 이 모델은 시냅스 행렬에 저장된 벡터내에서 1,0의 수가 같다는 가정에서 출발하였다.[1,3] 그러나, 실제 정보들은 벡터간에 1의 수가 크게 차이나거나 또는 벡터내에서 1,0의 갯수가 다를 수도 있으며, 이 경우에는 이 모델의 가장 비슷한 것을 찾아가는 특성이 아주 나빠진다. 다시 말해서, 뉴런들의 상태가 이렇게 2진수로 표시될 때, 벡터내의 1,0수가 같은 경우에는 뉴런들이 기억된 벡터 중에서 입력과 가장 비슷한 것을 찾아가는 특성이 좋지만, 그렇지 않은 경우에는 여러 문제점들이 나타난다. Hopfield 모델의 이러한 문제점들을 알아보면 다음과 같다.

첫째, 기억된 벡터간에 1의 갯수가 차이날 경우, 입력에 가장 비슷한 것이 항상 최후 안정 벡터가 되지 않고, 기억된 벡터 중에서 1의 수가 많은 것이 1의 수가 적은 것보다 최후 안정 벡터가 되는데 어느 정도의 우위성을 가진다.

둘째, 기억된 벡터 중에서 1의 수가 다른 벡터보다 너무 적은 벡터는 비록 자신을 입력으로 하더라도 그 자체가 최후 안정 벡터가 되지 못한다.

세째, 기억된 벡터간에는 1의 수가 같더라도 기억된 벡터내에서 1,0의 수가 크게 차이날 경우이다. 기억된 벡터내에서 1의 수가 0의 수에 비해 너무 적으면, 기억된 벡터들의 몇 번째 요소는 모두 0이더라도 여기에 해당하는 행렬 T의 행 요소(row element)가 모두 양수여서, 출력의 요소중 여기에 해당하는 비트 값이 항상 1이 나온다. 1의 수가 0의 수에 비해 훨씬 많은 벡터들이 기억된 경우에는 앞과 역으로 된 관계가 나타난다. 이 두 경우에도 기억된 벡터가 자신을 입력으로 하였지만 그 자체가 최후 안정 벡터가 되지 못한다.

표1과 표2는 Hopfield 모델의 이런 문제점을 보여주는 입력이다. 표1의 (a)에 주어진 3개의 벡터가 시냅스 행렬에 기억되었을 때, 표1.(b)에서와 같이 입력이 주어지면, 입력이 A와 10비트, B와 11비트, C와 14비트가 같으므로, 입력과 가장 비슷한 벡터인 C가 출력으로 예상된다. 그러나 이 경우에 1의 수가 다른 것보다 비교적 많은 A가, 비록 입력과 가장 비슷한 벡터는 아니지만, 최후 안정 벡터가 된다. 즉, 1의 수가 많은 A의 우위성이 나타난다. 또, 표1의 (c)는, 1의 수가 6인 C가 입력이 되어도 그 자체가 최후 안정 벡터가 되지 못함을 보여준다. 이것은 C를 시냅스 행렬에 기억시켰더라도 실제는 기억되지 않은 것을 나타낸다. 표2는 세번째 문제점에 해당하는 예로써, 표2.(a)에서 기억된 벡터들을 보면 1,5,9,13,17번째 요소가 모두 0이다. 이때는 표2.(b)에서 보는 바와 같이 여기에 해당하는 시냅스 행렬의 행은 그 요소가 모두 양수여서, 전부 0가 아닌

표 1. 기억된 벡터간에 1의 수가 차이날 경우의 계산 결과
(a)기억된 벡터. (b) 1의 수가 많은 벡터의 우위성이 나타남. (c) C가 최후 안정 벡터가 안됨.
(No(1):벡터내에 있는 1의 수 SB:두 벡터간에 같은 비트 수)

```
(a) Three Memorized Words  No.(1)        SB
    A:111100001111000001111     12    A-B: 11
    B:101010001110110010100      9    B-C: 15
    C:101000001100010001001      6    C-A: 12

                                    A   B   C
(b) INTPUT VECTOR
    110000000000000000000 SB: 10  11  14
    FINAL STABLE VECTOR
    111100001111000001111 SB: 20  11  12

(c) INTPUT VECTOR
    101000001000100001001 SB: 12  15  20
    FINAL STABLE VECTOR
    101100000111010001001 SB: 14  17  19
```

표 2. 기억된 벡터내에서 1,0수가 차이날 경우의 계산 결과
(a) 기억된 벡터. (b) 시냅스 행렬. (c) 잘못된 안정 벡터.

```
(a) Three Memorized Words  No.(1)        SB
    A:00010001000100000100010     5    A-B: 10
    B:10010011000001100100100     5    B-C: 10
    C:01000000100110011000011     5    C-A: 10

(b) 0  1  1  1  3  1  1  1  3  1  1  1  3  1  1  1  3  1  1  1
    1  0 -1 -1  1 -1 -1  1  3  1 -1  1  3 -1 -1  1  3 -1 -1  1
    1 -1  0 -1  1 -3  1 -1  1  3 -1  1 -3  1 -1 -1  1 -3  1 -1
    1 -1 -1  0  1 -1  3 -1  1  3 -1 -1  1 -3  1 -1  1 -3  1 -1
    3  1  1  1  0  1  1  1  3  1  1  1  3  1  1  1  3  1  1  1
    1 -1  3 -1  1  0 -1 -1  1  3 -1  1  3 -1  1  3 -1 -1  1  3
    1 -1 -1  3  1 -1  0 -1  1  3 -1 -1  1 -3  1 -1  1 -3  1 -1
    1  3 -1 -1  1 -1 -1  0  1 -3  1  3 -1 -1  1 -3  1 -1 -1  1
    3  1  1  1  3  1  1  1  0  1  1  1  3  1  1  1  3  1  1  1
    1 -1 -1  3  1 -1  3 -1  1  0 -1 -1  1 -3  1 -1  1 -3  1 -1
    1  3 -1 -1  1 -1 -1  3  1 -1  0 -1  1  3 -1 -1  1 -3  1 -1
    1 -1  3 -1  1  3 -1 -1  1 -1 -1  0  1 -3  1  3 -1 -1  1 -3
    3  1  1  1  3  1  1  1  3  1  1  1  0  1  1  1  3  1  1  1
    1  3 -1 -1  1 -1 -1  3  1 -1  3 -1  1  0 -1 -1  1 -1 -1  3
    1 -1  3 -1  1  3 -1 -1  1  3 -1 -1  1  0 -1  1  3 -1 -1  1
    1 -1 -1  3  1 -1  3 -1  1  3 -1 -1  1 -3  0  1  1  3 -1 -1
    3  1  1  1  3  1  1  1  3  1  1  1  3  1  1  1  0  1  1  1
    1 -1  3 -1  1  3 -1 -1  1  3 -1  3 -1  1  1  0 -1 -1  1 -1
    1 -1 -1  3  1 -1  3 -1  1  3 -1 -1  1 -3  1  1  0 -1  1 -1
    1  3 -1 -1  1 -1 -1  3  1 -1  3 -1  1  3 -1 -1  1 -3  1  0

                                    A   B   C
(c) INTPUT VECTOR
    00010001000000010010 SB: 20  10  10
    FINAL STABLE VECTOR
    10011010110010011010 SB: 15   5   5
```

어떤 입력을 가지고 계산을 하더라도 i=1,5,9,13,17인 u_i은 항상 양수가 되고, 그래서 v_i의 다음 상태는 항상 1이 된다. 결국, 기억된 것 중에 어느 것을 입력으로 하든지 출력은 기억된 것과 5비트가 틀린 잘못된 안정 벡터인 것을 표2.(c)에서 볼 수 있다.

4. 알고리즘의 수정

3절에서 보였던 문제점은 모두 기억된 벡터에서 1,0갯수의 불균형 때문에 생기는 문제이다. 앞에서 언급했던 것처럼 이 모델은 1,0의 수가 같거나 비슷할 경우에만 적용이 가능하므로, 벡터(1)에 보수 관계의 벡터를 붙여주어 길이는 2배 확

장되고 1,0의 수는 같은 벡터를 만들어 주자. 확장된 벡터를 $U^{(m)}$이라 하면 벡터(1)과의 관계는 다음과 같다.

$$U^{(m)} = (u_1^{(m)}, u_2^{(m)}, \cdots, u_{2N}^{(m)}), \quad m = 1, 2, \cdots, M \qquad (5)$$

$$u_\mu^{(m)} = \begin{cases} v_\mu^{(m)} & 1 \le \mu \le N \\ 1 - v_{\mu-N}^{(m)} & N+1 \le \mu \le 2N \end{cases} \qquad (6)$$

이 벡터(5)를 저장하는 시냅스 행렬을 $T'(2N \times 2N)$이라 하자. 그 요소 $t'_{\mu\nu}$는 다음과 같다.

$$t'_{\mu\nu} = \sum_{m=1}^{M} (2u_\mu^{(m)} - 1)(2u_\nu^{(m)} - 1) - M\delta[\mu - \nu] \qquad (7)$$

이때, 벡터(5)는 벡터(1)을 코딩한 것이므로 벡터(5)의 시냅스 행렬 T'의 요소 $t'_{\mu\nu}$는, (7)식에 (6)식을 대입하면, 벡터(1)의 시냅스 행렬 T의 요소 t_{ij}로 나타낼 수 있을 것이다. 이 두 행렬 T'과 T사이의 관계를 알아보기 위해 행렬 T'을 그림1.처럼 4개의 영역으로 나누자.

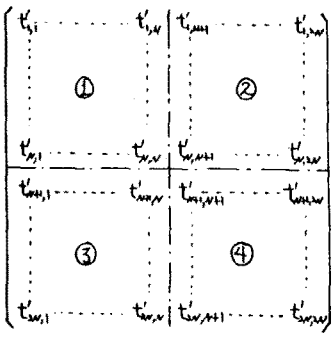

그림 1. 시냅스 행렬 T'의 영역 구분

먼저, 영역①($1 \le \mu, \nu \le N$)에서 T'의 요소 $t'_{\mu\nu}$는, (7)식에 (6)식을 대입하고 (2)식을 이용하면 다음과 같다.

$$t'_{\mu\nu} = t_{\mu\nu}, \quad 1 \le \mu, \nu \le N \qquad (8)$$

그다음, 영역②($1 \le \mu \le N, N+1 \le \nu \le 2N$)에서 $t'_{\mu\nu}$는 위와 같은 방법으로 (7)식에 (6)식을 대입하면 다음과 같다.

$$t'_{\mu\nu} = -\sum_{m=1}^{M} (2v_\mu^{(m)} - 1)(2v_{\nu-N}^{(m)} - 1) - M\delta[\mu - \nu] \qquad (9)$$

여기에, (2)식의 관계를 이용하면 $t'_{\mu\nu}$는 아래와 같다.

$$t'_{\mu\nu} = \begin{cases} -t_{\mu,\nu-N} & \mu \neq \nu - N \\ -M & \mu = \nu - N \end{cases} \qquad (10)$$

영역③($N+1 \le \mu \le 2N, 1 \le \nu \le N$)에서도 역시 위와 같이 하면 $t'_{\mu\nu}$는

$$t'_{\mu\nu} = \begin{cases} -t_{\mu-N,\nu} & \mu - N \neq \nu \\ -M & \mu - N = \nu \end{cases} \qquad (11)$$

이다.

마찬가지로, 영역④($N+1 \le \mu, \nu \le 2N$)에서 $t'_{\mu\nu}$는 다음과 같다.

$$t'_{\mu\nu} = t_{\mu-N,\nu-N}, \quad N+1 \le \mu, \nu \le 2N \qquad (12)$$

따라서, 벡터(6)의 시냅스 행렬 T'의 모든 요소 $t'_{\mu\nu}$는 위과 같이 행렬 T의 요소로 표시할 수 있으며, 입력 벡터 V가 (6)식과 같이 U로 코딩되었을 경우의 계산 알고리즘은 다음과 같다.

$$\hat{u_\mu} = \sum_{\nu=1}^{2N} t'_{\mu\nu} u_\nu = \begin{cases} \sum_{\nu=1}^{N} t_{\mu\nu}(2v_\nu - 1) + M(v_\mu - 1), & 1 \le \mu \le N \\ -\sum_{\nu=1}^{N} t_{\mu-N,\nu}(2v_\nu - 1) - Mv_{\mu-N}, & N+1 \le \mu \le 2N \end{cases} \qquad (13)$$

이 $\hat{u_\mu}$에서 다음과 같이 u_μ의 다음 상태가 결정된다.

$$u_\mu = TH\{\hat{u_\mu}\} = \begin{cases} 1 & \hat{u_\mu} > 0 \\ 0 & \hat{u_\mu} \le 0 \end{cases} \qquad (14)$$

이때, (7)식에서 $1 \le \mu \le N$ 인 경우 $u_\mu = v_\mu$인 것을 알 수 있다. 그러므로, 계산 알고리즘을 식(3)에서 (13)으로 바꾸어주면, 실제로 벡터를 코딩하고 4배 크게 할 필요없이, 모든 벡터가 1,0의 수가 같은 벡터로 코딩된 것과 같이 처리되었으므로 1,0의 갯수에 의해 야기된 문제점은 해결된다.

표3은 표1,2와 같은 벡터와 시냅스 행렬에 새 알고리즘을 적용시킨 결과이며, 이 표 3개를 비교해 보면 1,0의 갯수 때문에 나타난 문제가 해결된 것을 알 수 있다. 표1.(c)에서 보는 바와 같이 C는 자신을 입력으로 하여도 출력이 되지 않았지만, 표3.(a)에서, 새 알고리즘에 따라 벡터 상태의 시간적 변동이 진전될 때, 1의 갯수가 6개 뿐인 C가 입력과 6비트 다른데도 입력과 가장 비슷한 벡터이기 때문에 최후 안정 벡터가 된 것을 볼 수 있다. 표2.에서는 i=1, 5,9,13,17일 때, u_i의 다음 상태 값이 항상 1이 되었는데, 표3.(b)에서 보는 바와 같이 이 문제점이 고쳐져서 입력과 가장 비슷한 벡터 A가 최후 안정 벡터가 되었다.

표 3. 표1,2와 같은 벡터에 새 알고리즘 적용 결과
(a) 표1의 단점 해결. (b) 표2의 단점 해결.

```
                                                    A  B  C
(a)  INTPUT VECTOR
     1000000000000000000000 SB: 10 11 14
     FINAL STABLE VECTOR
     1010000010001000010011 SB: 12 15 20

                                                    A  B  C
(b)  INTPUT VECTOR
     0001001000000000000000 SB: 17 13 13
     FINAL STABLE VECTOR
     0001001001000001001010 SB: 20 10 10
```

5. 결 론

이 논문에서 Hopfield 모델에 나타나는 몇 개의 문제점을 그 원인과 함께 보이고, 해결책을 제시하였으며, 이를 통하여 문제점이 해결된 것을 보였다.

여기서, 해결책으로 제시한 방법은 계산 알고리즘을 수정한 것이므로, 벡터의 코딩이나 기억 용량(memory space) 즉, 시냅스 행렬(synaptic matrix)의 확장없이 1,0 갯수의 불균형 때문에 야기되는 문제점을 해결하였다.

-참고 문헌-

1. J. J. Hopfield, "Neural Networks and Physical Systems with Emergent Collective Computational Abilities," Proc. Natl. Acad. Sci. 79, 2554 (1982)

2. J. J. Hopfield, "Neurons with Graded Response have Collective Computational Properties Like Those of Two-State Neurons," Proc. Natl. Acad. Sci. 81, 3088 (1984)

3. D. Psaltis and N. Farhat, "Optical information processing based on associative-memory model of neural nets with thresholding and feedback," Opt. Lett. 10, 98 (1985)

4. N. Farhat, D. Psaltis, A.Prata, and E. Paek, "Optical information processing of the Hopfield model," Appl. Opt. 24, 1469 (1985)

5. B. Macukow and H. H. Arsenault, "Modification of the threshold condition for a content-addressable memory based on the Hopfield model," Appl. Opt. 26, 34 (1987)