정보학연구 제7권 제2호 2004년 6월(pp.43-54)
The Journal of Information Technology, Vol.7, No.2

선형 선처리 방식에 의한 홉필드 네트웍의 성능 분석

고영훈*, 이수종**, 노홍식***

요 약

홉필드 네트웍(Hopfield Network)은 존 홉필드(John J. Hopfield) 박사에 의해 제안된 이래 패턴인식과 최적화 문제에 활용되어 왔다. 특히 리(Jian-Hua Li)에 의해 제안된 방식은 SVD(singular value decomposition) 기법을 사용하여 입력패턴을 재구성함으로써 효율향상에 기여하였다.

본 논문은 리(Li)가 제안한 홉필드 네트웍에 사용할 패턴 집합의 선형 선처리 방식에 따른 성능 향상을 실험하였다. 선형 선처리 방식에 하다마드 방식과 랜덤 방식의 두 가지 방식을 사용하였으며 두 방식 모두 성능향상이 있음을 확인하였다. 성공률 측면에서 보면 랜덤 방식이 최대 30%, 하다마드 방식이 최대 15%의 성능이 향상되었다. 수렴시간 측면에서 보면 랜덤 방식이 최대 5 이터레이션, 하다마드 방식이 최대 2.5 이터레이션의 성능 향상을 확인하였다.

Ⅰ. 서론

홉필드 네트웍이 존 홉필드에 의해 제안된 이후 크게 두가지 영역에서 기존 시스템과 경쟁하게 되었으며, 프로세서의 발달로 실시간 처리가 가능한 뉴런의 수가 증가할수록 월등한 성능을 발휘할 것이다. 두가지 영역은 패턴 매칭과 경로 최적화이다[1][2][3]. 임의의 손상된 입력 패턴에 의해서 홉필드 네트웍이 수렴하면 이미 입력된 패턴 집합 중에서 가장 가까운 패턴을 찾아주는 것이 패턴 매칭이다. 그리고 네트웍 상의 한 노드에서 목적지 노드까지 도달하는 가장 최적화된 경로를 찾아주는 것이 경로 최적화이다.

두 영역 모두 기본적인 동작 원리가 리야프노프(Lyapunov) 함수의 수렴과정에 대한 수학적 모델로 귀결된다. 두 영역의 차이점은 패턴 매칭에서의 뉴런은 이미지를 의미하지만, 경로 최적화에서의 뉴런은 경로를 의미한다는 차이점이 있다[4][5].

리야프노프 함수에 의해 효과적으로 수렴되기 위해서는 기입력된 패턴 집합의 스페이스 분포가 상당히 중요하다. 패턴 간의 해밍거리(Hamming distance)가 가깝지 않아야 하기 때문이다. 임의의 패턴 집합에서 적절한 해밍거리를 유지한다는 것을 항상 기대하기는 쉽지 않다. 따라서 리(Li)는 SVD(singular value decomposition) 기법을 이용한 패턴 재배치 방식을 사용하여 패턴간에 적절한 해밍 거리를 확보하고 있다[6].

그럼에도 불구하고 여전히 선처리 과정에 의한 패턴 재배치는 유효하며 어느정도의 성능향상을 보이고 있다. 본 논문에서는 선형 선처리 과정의 두가지 방식을 제시하여 수렴 성공률과 수렴 시간에 대한 폭넓은 실험을 통하여 구체적인 성능향상의 정도를 알아보고자 한다.

뉴런의 개수가 N이라면 가능한 뉴런의 패턴

* 협성대학교 컴퓨터공학과 교수
** 협성대학교 컴퓨터공학과 조교수
*** 협성대학교 컴퓨터공학과 교수

개수는 2^N으로 주어질 수 있으며, 이 중에서 직교 패턴의 개수는 $2^{\sqrt{N}}$승으로 가능한 패턴의 개수에 비해 상당히 적은 부분을 차지하고 있다. 하다마드 패턴 집합은 이상적이 경우로 직표 패턴의 각 패턴간 해밍거리가 모두 같은 뿐만 아니라 가장 길게 배치된 경우이다. 랜덤 패턴 집합의 경우도 거의 비슷한 해밍거리를 얻을 수 있지만 하다마드 패턴 집합의 경우보다는 못하다.

해밍거리가 긴 이 두가지 패턴 집합을 얻기 위한 선형 선처리 방식을 사용하여 성능향상이 있는지의 여부를 알아본다. 성능 향상이 있다면, 각 방식별로 어느정도의 성능향상이 있는지, 반복된 시뮬레이션을 통하여 신뢰성 있는 결과를 유추하고자 한다.

II. 홉필드 네트웍

컴퓨터와 뇌를 비교해 보면 기본적으로 정보처리의 방식이 전혀 다르다. 컴퓨터는 주어진 프로그램에 따라서 한 번에 하나의 명령을 처리하는 직렬처리를 하는 반면, 뇌는 수많은 뉴런들이 병렬처리를 한다. 그리고 컴퓨터는 문제를 풀기 위하여 문제의 해결 과정, 즉 알고리즘을 컴퓨터 언어로 구성하여야 하는 반면에 뇌는 학습에 의하여 문제를 해결하게 된다. 그 결과 뇌는 학습한 문제뿐만 아니라 학습하지 않은 문제에 대하여 유추를 통하여 해결할 수 있다. 이러한 뇌의 특성, 즉 신경망을 모방한 것이 뉴럴 네트웍(인공신경회로망)이다.

홉필드 네트웍 모델은 1982년 미국 캘리포니아(Caltech Univ.) 공대 교수인 홉필드(John J. Hopfield)에 의해 제안된 것으로 연상 기억이나 최적화를 다루는 문제에 많이 이용된다. 연상 기억에 있어서는 일정한 패턴들을 계산하여 미리 연결 강도로 저장하였다가, 미지의 입력 패턴이 들어오면 유사한 패턴을 찾아내는 방식을 사용한다. 그리고 최적화를 다루는 대표적인 문제로 순회판매원(Traveling Salesman Problem) 문제가 널리 알려져 있다. 이는 최소의 비용으로 목적하는 곳을 한번씩 방문하는 해를 구하는 것인데, 이는 최적 라우팅의 문제로도 많이 연구되고 있다.

홉필드 네트웍은 모든 뉴론은 결합강도(w_{ij})에 의해 대칭적으로 완전 결합되어 있다. 또한 각 뉴론의 값은 -1에서 +1의 범위로 제한되어 있으며, 기억 패턴의 값은 -1과 +1의 두 개의 값만 가질 수 있다.

이러한 구조에서 뉴런의 초기값을 입력하면 리야프노프 함수가 지속적으로 감소하면서 뉴런의 값이 변화하게 된다. 결국 리야프노프 함수의 최소값으로 뉴런이 수렴하게 되는데, 이 뉴런의 값은 초기값과 가장 가까운 패턴을 가리키게 된다.

1982년 홉필드에 의해 발표된 "Neural networks and physical systems with emergent collective computational abilities" 란 간결한 논문은 신경망 연구에 큰 영향을 끼쳤다. 이는 홉필드가 상당히 유명한 대학(Caltech Univ.) 및 연구 기관(Bell Labs)과 연관이 있는 저명한 물리학자였다는 점이 작용하였기 때문이다.

일본의 아마리(Amari)는 1972년 신경망의 역동을 분석하는데 있어서 일찌기 리야프노프(Lyapunov) 함수를 강조했으며, 허멜(Hummel)과 쥬커(Zucker)는 1981년에 "대칭연결 네트워크는 매우 중요하고도 특수한 경우이며, 그들의 행위는 에너지 함수에 의해 지배된다."라는 요지의 기술적인 리포트를 하였으며 이에 관한 최종

리포트는 1983년에 발표되었다.

홉필드 네트워크와 유사하면서도 두 가지 면에서 다른 모델은 마르(David Marr) 와 포지오(Poggio)가 제안하였다. 그러나 에너지 함수의 사용이 곧 신경망 행위를 분석하고 제어한다는 것을 주장한 사람은 홉필드였다.

홉필드 네트웍의 각 뉴런을 벡터(x)로 뉴런간의 연결강도를 매트릭스(W)로 표현하면 홉필드 네트웍을 선형변환과 비선형함수의 결합된 매트릭스 수식으로 표현할 수 있다. 연결강도를 구성하는 방법은 여러가지가 있을 수 있다. 기본적인 방법은 N_P개의 패턴 벡터들의 선형결합에 의해 연결강도를 결정하는 방법이다. 일반적으로 N_P의 값은 뉴런의 개수인 N_N보다 작으며, $N_N/5$ 보다 적은 경우에 해가 잘 구해지는 것으로 알려져 있다.

연결강도는 그 자체로 모든 저장된 패턴들을 반영하고 있으며, 입력 패턴에 따라서 가장 유사한 패턴으로 수렴된다. 이는 직교집합에서 입력값에 따라 직교집합의 한 원소만 추출되는 원리와 유사하다.

그러나 문제는 저장된 패턴들의 직교성은 거의 기대할 수 없으므로, 개선된 방법으로 연결강도를 구한다. 리(Li)에 의해 체계화된 이 방법은 연결강도를 구하기 위하여 저장될 패턴들의 선처리(pre-processing) 과정을 수행하여 최대한 직교성을 확보하는 것이 요점이다. 연결강도를 구할 때 가장 중요한 요점은 저장될 임의의 패턴들간의 거리를 최대한 분산시켜 놓는 것이다. 이를 위해서는 패턴보다 패턴들 사이의 거리가 더 중요한 정보가 된다. 즉, 패턴간 거리 벡터를 매트릭스화하여 아이겐밸류를 찾아내는 SVD (singular value decomposition)를 이용하여 재배치된 벡터를 사용하여 연결강도를 구하고 있다.

Ⅲ. 홉필드 네트웍의 선처리 방식

홉필드 네트웍의 기본 성능은 저장된 패턴의 직교성(orthogonality)에 의해서 결정된다. 즉, 최상의 성능을 내기 위해서는 각각의 패턴집합이 직교집합으로 형성되어 있을 경우이다. 직교집합의 대표적인 예인 하다마드 집합을 사용하였을 경우가 그 예이다. 직교집합보다는 성능이 떨어지지만 그와 유사한 성능을 보이는 패턴집합은 랜덤패턴의 경우이다. 이 경우는 랜덤한 특성 때문에 패턴간의 상관관계(correlation)가 매우 적기 때문이다. 대부분의 경우에는 패턴집합의 상관관계가 어느 정도 존재하며 이를 낮추기 위해서는 패턴집합의 인위적인 랜덤화 과정을 사용하는 것이다. 이때 사용하는 랜덤화 매트릭스는 역함수가 가능한 매트릭스로 상호변환이 가능하다.

3.1. 패턴 집합의 고찰

홉필드 네트웍의 중요한 파라메타는 뉴런의 개수이다. 뉴런의 개수가 증가하면 뉴런간 연결로도 지수적으로 증가하며 패턴의 개수도 지수적으로 증가하게 된다. 계산의 명확성을 위해 다음의 파라메타를 정의하였다.

패턴 매칭을 위해서는 뉴런이 2차원 이미지에 대응되어야 하므로 본 논문에서는 8, 10, 12의 인덱스를 대상으로 하였다. 인덱스가 8인 경우 뉴런의 개수는 256개이며 16x16의 이미지에 대응된다.

각 인덱스의 직교율(rate of orthogonality)과 패턴사용율(rate of using)을 살펴보면 다음과 같다.

〈표 1〉 파라메타 목록

명칭	영문	기호 및 수식
인덱스	index	I
뉴런의 개수	Number of Neurons	$N_N = 2^I$
전체 패턴 개수	Number of pattern Spaces	$N_S = 2^{N_N} = 2^{2^I}$
모사 직교 패턴 개수	Number of pattern Orthogonals	$N_O = N_N = 2^I$
패턴 개수	Number of Patterns	$N_P = 10(this\ paper)$
직교율	Rate of Orthogonality	$R_O = \frac{N_O}{N_S} = \frac{2^I}{2^{2^I}} = 2^{(I-2^I)}$
사용율	Rate of patter Using	$R_U = \frac{N_P}{N_O} = \frac{10}{2^I}$

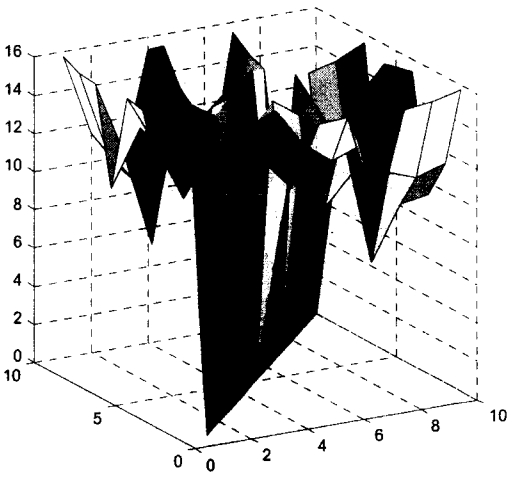
〈표 2〉 인덱스별 직교율과 패턴사용율

인덱스(I)	직교율	패턴사용율	폰트사이즈
8	2.21E-75	3.91%	16x16
10	5.69-306	0.98%	32x32
12	≒0	0.24%	64x64

직교율은 전체 패턴의 개수에서 직교 패턴의 개수의 비율을 나타낸 것으로 인덱스가 증가할수록 그 값은 0에 근접한다. 실질적으로 사용가능한 패턴의 개수는 직교패턴의 개수를 넘을 수 없다. 패턴 사용율은 그 직교패턴 중에서 실제 사용하는 패턴(10개)의 비율을 나타낸 것으로 인덱스가 8인 경우에 3.91%의 사용율을 보인다.

실험에 사용될 패턴은 기본적인 숫자 패턴으로 쉽게 얻을 수 있는 윈도우 폰트를 사용하였다. 그림 4는 각 패턴간의 해밍거리를 나타낸 것이다. X축과 Y축이 같은 경우는 동일 패턴의 해밍거리임으로 0을 가리키게 된다. 대부분의 패턴간 해밍거리는 일정 구간에 위치하게 되는데, 평균 해밍거리와 최소 해밍거리는 패턴집합의 중요한 특징이 된다.

최적화된 패턴 집합이란 동일한 조건에서 평균 해밍거리가 가장 길고, 최소 해밍거리 역시 평균 해밍거리와 같아야 한다. 이러한 조건을



(그림 1) 패턴간 해밍거리

가진 패턴 집합은 하다마드 패턴 집합이 있다. 하다마드 패턴 집합보다는 평균 해밍거리가 짧고, 최소 해밍거리도 평균 해밍거리보다 짧지만, 거의 유사한 특성을 가지고 있는 것이 랜덤 패턴 집합이다. 구체적인 수치를 도출하기 위해 우선 5가지 패턴 집합을 정의하고, 집합내 패턴간 평균거리와 최소거리를 계산한다.

기본적으로 해밍거리 특성이 우수한 하다마드 패턴 집합(pl), 비슷한 특성을 보이는 랜덤 패턴

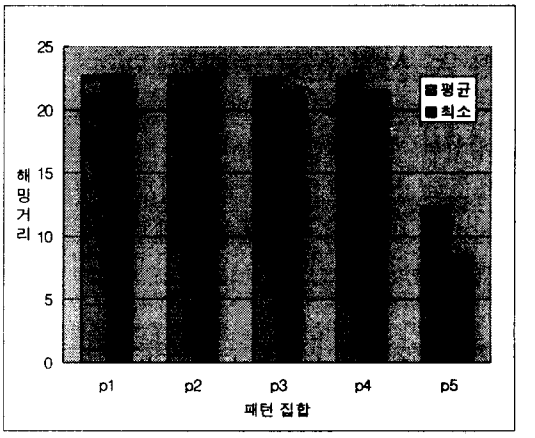
〈표 3〉 패턴 집합의 목록

인덱스	내용	개수
p1	하다마드 패턴 집합	N_N
p2	랜덤 하다마드 패턴 집합	N_N
p3	랜덤 패턴 집합	N_N
p4	랜덤 패턴 집합	N_P
p5	숫자 패턴 집합	N_P

집합(p3) 및 임의 패턴 집합인 숫자 패턴 집합(p5)를 사용하였다. 하다마드 패턴들은 그 값들이 노출되어 있으므로 임의의 랜덤패턴과 하다마트 패턴의 곱으로 형성된 새로운 랜덤(하다마드) 패턴 집합(p2)를 사용하여 하다마드 패턴 집합(p1)과 비교하여 보았다. 또한 패턴의 개수가 N_N인 랜덤 패턴 집합(p3)와 패턴의 개수가 N_P인 랜덤 패턴 집합(p4)도 비교하여 보았다.

<표 1>, <표 2>, <표 3>은 각각의 패턴집합에 대한 평균 해밍거리와 최소 해밍거리에 대한 값을 보여준다. <표 1>에서는 16x16 사이즈의 폰트로 보통 굴림체 및 폰트 크기는 12를 사용하였다. <표 2>에서는 32x32 사이즈의 폰트로 보통 굴림체 및 폰트 크기는 24를 사용하였다. <표 3>에서는 64x64 사이즈의 폰트로 보통 굴림체 및 폰트 크기는 48를 사용하였다.

(그림 5)에는 256개의 뉴런을 사용했을 경우의 각 패턴 집합에 대한 최소 해밍거리와 평균



(그림 2) 패턴집합에 대한 평균거리 및 최소거리

〈표 4〉 패턴별 해밍거리(I=8, N_N= 256, N_P =10)

	하다마드	랜덤(하다마드)	랜덤	랜덤(10)	패턴(10)
평균 해밍거리	22.627	22.627	22.452	22.589	12.284
최소 해밍거리	22.627	22.627	21.549	21.396	8.460

〈표 5〉 패턴별 해밍거리(I=10, N_N= 1024, N_P =10)

	하다마드	랜덤(하다마드)	랜덤	랜덤(10)	패턴(10)
평균 해밍거리	45.255	45.255	45.242	45.184	21.482
최소 해밍거리	45.255	45.255	44.205	44.065	16.765

〈표 6〉 패턴별 해밍거리(I=12, N_N= 4096, N_P =10)

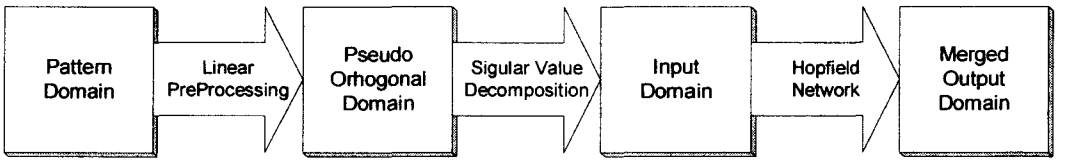
	하다마드	랜덤(하다마드)	랜덤	랜덤(10)	패턴(10)
평균 해밍거리	90.5097	90.5097	90.4116	90.2009	42.3757
최소 해밍거리	90.5097	90.5097	89.3885	88.8364	33.6235

해밍거리에 대한 그래프를 나타내었다.

고찰 결과 4개의 패턴 집합(p1,p2,p3,p4)은 유사한 특성을 보인 반면에 숫자 패턴 집합(p5)는 상대적으로 거리특성이 50%정도로 열악한 것을 알 수 있다. 따라서 임의의 패턴 집합을 하다마드 패턴 집합이나 랜덤 패턴 집합으로 선형 변환할 수 있다면 홉필드 네트웍의 성능향상을 기대할 수 있다.

3.2. 선형 선처리 방식

임의의 패턴 집합을 랜덤 또는 하다마드 패턴 집합으로 선형 변환하기 위한 방법을 알아본다. 그림 3은 선형 선처리 홉필드 네트웍의 플로우그램을 나타내고 있다.



(그림 3.)선형선처리 방식의 홉필드 네트워크 플로우그램

패턴 도메인에서 모의 직교 도메인으로 선형 변환하기 위해서 선형 선처리 행렬을 생성한다. 변환행렬의 특성상 정방행렬이어야 하므로 패턴의 개수를 임의 조정하여 변환행렬을 생성한다. 일단 선형 선처리 행렬이 생성되면 패턴 도메인에 있는 임의의 패턴을 선형 선처리 행렬을 통하여 모의 직교 도메인에 있는 패턴으로 변환할 수 있다.

선형 선처리 행렬은 모의 직교 도메인의 종류에 따라 하다마드 및 랜덤의 두가지로 구분된다. 다음 수식은 하다마드 모의 직교 도메인으로 변환하기 위한 선형 선처리 행렬을 구하기 위한 것이다. 패턴 세트 행렬(P)를 중간 단계(Intermediat)의 패턴 세트 행렬(P_I)로 변환한 후에 변환 행렬(A_H)을 곱하여 하다마드 행렬(H)을 얻게 된다. 구하고자 하는 변환 행렬을 얻기 위해서 중간 단계의 패턴 집합 행렬의 역행렬을 사용한다.

$$\begin{aligned} A_H \cdot P_I &= H \\ A_H \cdot P_I \cdot P_I^{-1} &= H \cdot P_I^{-1} \\ A_H &= H \cdot P_I^{-1} \end{aligned} \quad (1)$$

일단 하다마드 선형 선처리 행렬(A_H)이 구해졌으면 간단한 행렬 곱셈을 통하여 모의 직교 패턴을 구할 수 있다.

$$A_H \cdot P_{(i)} = P_{\alpha(i)} \quad (2)$$

랜덤 선형 선처리 행렬(A_R)도 같은 과정에 의해서 구할 수 있다.

$$\begin{aligned} A_R \cdot P_I &= R \\ A_R \cdot P_I \cdot P_I^{-1} &= R \cdot P_I^{-1} \\ A_R &= R \cdot P_I^{-1} \end{aligned} \quad (3)$$

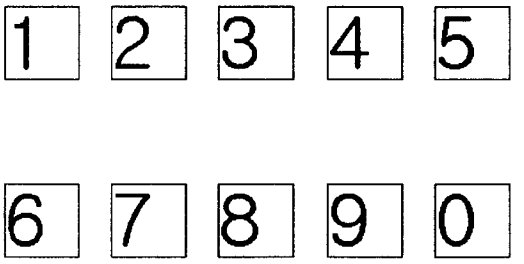
선형 선처리 행렬을 구하기 위해서는 중간 단계의 패턴 집합이 필요한데, 이것은 선형 선처리 행렬을 구하기 위한 역행렬 연산이 가능해야 되기 때문이다. 일반적으로 N_P는 N_N에 비해 매우 적으므로 ($N_N - N_P$)개에 해당되는 행(row)은 임의의 값으로 채운 후 선형 선처리 행

렬을 구하였다.

Ⅳ. 실험 및 결과

본 논문에서는 실험을 위한 시뮬레이션 툴로 매트랩(matlab)을 사용하였다. 뉴럴네트웍 툴박스에 있는 뉴럴홉필드(newhop) 함수를 중심으로 다수의 프로시져와 함수를 사용하여 시뮬레이션을 하였다. 신뢰성 있는 결과를 얻기 위해서 동일한 파라메타에서 최소 50번 이상의 시뮬레이션을 통한 평균값을 사용하였다.

실험을 위한 숫자 패턴집합은 1에서 0까지의 10개의 숫자패턴을 사용한다. 윈도우 시스템에서 사용하는 보통 굴림체 폰트를 이용하였다.



(그림 4) 숫자 패턴 집합

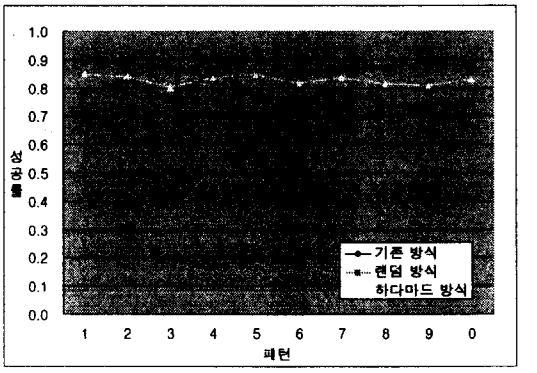
실험은 크게 성공률(success rate)와 수렴시간(success time)의 두가지 군으로 나누어 진행하였다. 실험에 사용된 패턴 집합은 숫자 패턴 집합, 랜덤 선형 선처리 패턴 집합, 하다마드 선형 선처리 패턴 집합의 세가지를 사용하여 비교하였다. 편의상 숫자 패턴 집합은 '기존 방식', 랜덤 선형 선처리 패턴 집합은 '랜덤 방식', 하다마드 선형 선처리 패턴 집합은 '하다마드 방식'으로 표기하였다.

홉필드 네트웍의 성능을 평가하기 위해서 입력 패턴에 잡음(noise)을 첨가하였으며, 잡음도(noise index)를 1에서 9까지 사용하였다. 즉 가우시안 분포하에서 범위가 -1에서 +1인 랜덤한 값을 발생시켜 패턴 이미지에 더하였다. 이때 이미지의 크기에 따라 잡음도를 부여하였는데, 잡음도가 1일 때 0.1에서부터 9일 때 0.9까지 증가하게 된다. 따라서 잡음도가 9인 경우는 잡음의 크기(magnitude)가 1이고 이미지의 크기가 0.9인 경우로 가장 잡음이 적게 섞인 경우가 된다.

4.1. 성공률 실험

성공률은 홉필드 네트웍의 이터레이션이 진행되면서 어느 정도 수렴에 성공하는지를 보여준다. 크게 패턴별 성공률과 잡음별 성공률에 대해 실험하였다.

홉필드 네트웍에서 이터레이션 20번 안에 수렴하면 성공으로 체크하였으며 50번 이상 반복실험하여 평균값을 기록하였다.

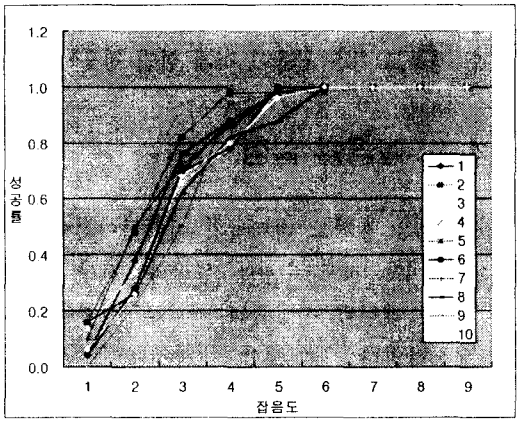


(그림 5) 선처리 방식에 대한 패턴별 성공률

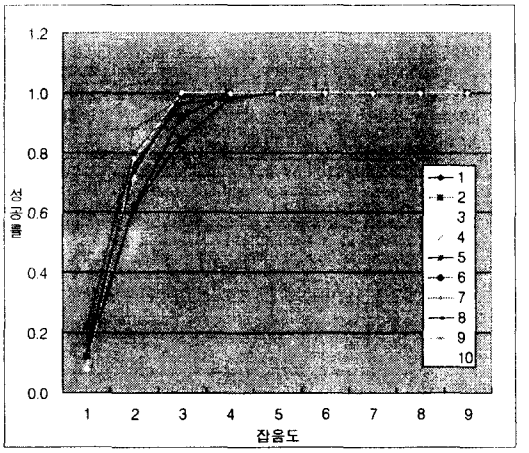
(그림 5)는 선형 선처리 방식에 대한 패턴별 성공률을 보여준다. 선형 선처리를 하지 않은 경우(노멀)와 랜덤 방식, 하다마드 방식을 쓴 경

우이다. 랜덤 방식, 하다마드 방식, 노멀의 순으로 성공률이 떨어지고 있는 것을 볼 수 있다. 대부분의 패턴에서 랜덤 방식이 우수하지만 특정패턴(4)에서는 오히려 성공률이 미세하지만 감소하는 것도 확인할 수 있다.

그림 6~8은 선형 선처리 방식에 대한 잡음별 성공률을 나타내고 있다.

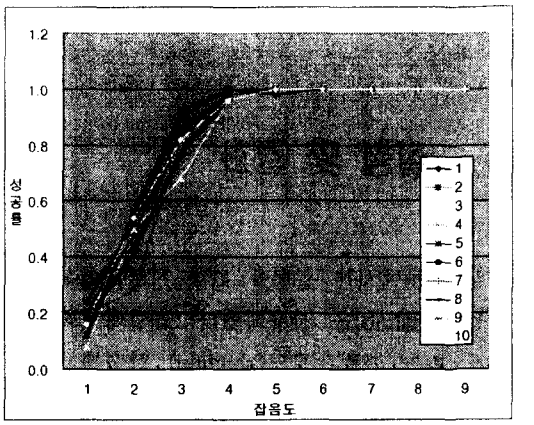


(그림 6) 기존 방식의 잡음별 성공률



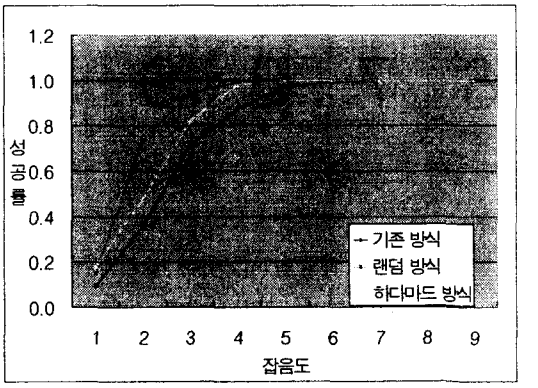
(그림 7) 랜덤 방식의 잡음별 성공률

(그림 12)에서는 각 패턴에 대해 평균값을 구한 후 잡음별 성공률을 표시하고 있다. 이 그래



(그림 8) 하다마드 방식의 잡음별 성공률

프에서는 각 선형 선처리 방식의 성공률을 상호 비교할 수 있다. 같은 노이즈 인덱스라면 랜덤 방식, 하다마드 방식, 기존 방식의 순으로 성공률이 나타났다. 노이즈 인덱스에 따라 성공률의 차이도 바뀌지만 하다마드 방식보다 랜덤 방식이 최대 15%, 기존 방식보다 하다마드 방식이 최대 15%의 성공률 향상이 있음을 알 수 있다.



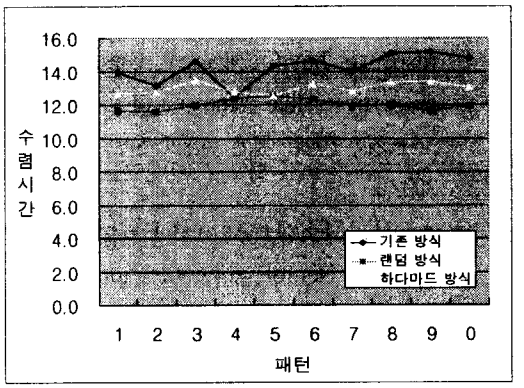
(그림 9) 선처리 방식에 대한 잡음별 성공률

4.2. 수렴시간 실험

홉필드 네트웍이 이터레이션을 진행하면서 수

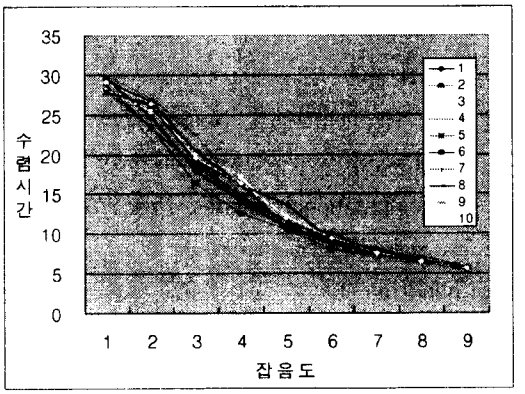
렴하기까지 걸리는 이터레이션 수를 측정하였다. 만약 20번의 이터레이션 안에 수렴에 실패한다면 최대 20번의 50%를 할증한 30번 이터레이션으로 간주하였다.

그림 10은 잡음 인덱스를 평균내어 선형 선처리 방식별에 대해 상호 비교할 수 있게 하였다. 랜덤 방식, 하다마드 방식, 기존 방식의 순으로 수렴시간이 길어지는 것을 알 수 있다.

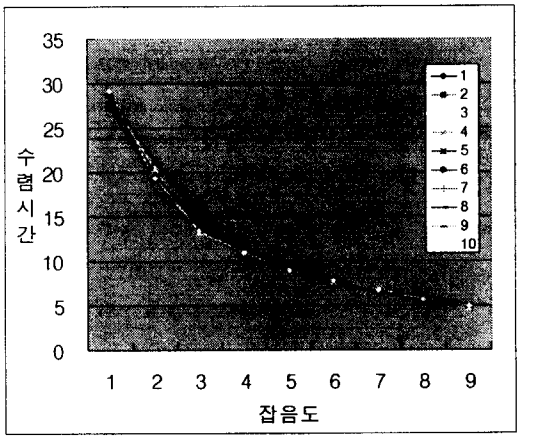


(그림 10) 선처리 방식에 대한 패턴별 수렴시간

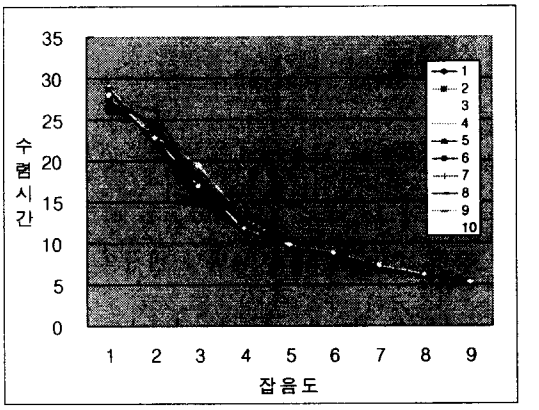
(그림 11~13)은 선형 선처리 방식에 대한 잡음별 수렴시간을 나타내고 있다. 잡음 인덱스가 증가할수록 수렴시간이 짧아지는 것을 확인할 수 있다.



(그림 11) 기본 방식의 잡음별 수렴시간

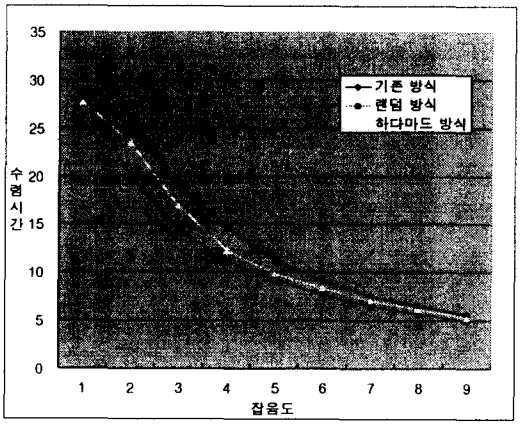


(그림 12) 랜덤 방식의 잡음별 수렴시간



(그림 13) 하다마드 방식의 잡음별 수렴시간

(그림 14)는 패턴에 대한 평균값을 취한 것으로 선형 선처리 방식에 대해 상호 비교할 수 있다. 잡음 인덱스에 따라 다르지만 랜덤 방식이 기존 방식보다 최대 5 이터레이션의 수렴시간이 빨라지는 것을 알 수 있다. 그리고 하다마드 방식은 기존 방식보다 최대 2.5 이터레이션 수렴시간이 빨라지는 것을 알 수 있다. 이러한 수렴시간의 단축은 잡음이 클수록 크게 나타나며, 잡음이 작은 경우에는 수렴시간의 차이가 거의 없음을 알 수 있다.



(그림 14) 선처리 방식에 대한 잡음별 수렴시간

V. 결론

홉필드 네트웍은 존 홉필드 박사에 의해 제안된 이래 패턴인식과 최적화 문제에 활용되어 왔다. 특히 리(Li)에 의해 제안된 방식은 SVD 기법을 사용하여 입력패턴을 재구성함으로써 효율향상에 기여하였다.

본 논문은 리(Li)가 제안한 홉필드 네트웍에 사용할 패턴 집합의 선처리 방식에 따른 실험을 하였으며, 선처리 방식에 따른 성능향상이 있음을 확인하였다. 선처리 방식은 패턴 집합의 평균거리 및 최소거리의 고찰 결과 하다마드 패턴 집합과 랜덤 패턴 집합의 거리 요소가 일반 패턴 집합의 거리 요소보다 우수한 점을 감안하여 두가지 방식을 선택하여 실험하였다.

실험결과 랜덤 방식이 하다마드 방식보다 우수하였다. 성공률 측면에서 보면 기존 방식보다 랜덤방식이 최대 30%, 하다마드 방식이 최대 15%의 성능이 향상되었다. 수렴시간 측면에서 보면 기존 방식보다 랜덤 방식이 최대 5 이터레이션, 하다마드 방식이 최대 2.5 이터레이션의 성능 향상을 확인할 수 있었다.

향후 과제로는 선형 변환을 위주로 한 선처리 방식 외에 비선형 변환인 치환 방식에서의 성능 분석이 요구된다.

참고문헌

[1] J. J. Hopfield(1984), Neurons with graded response have collective computational properties like those of two-state neurons, *Proc. Nat. Acad. Sci.*, 81, 3088-3092.

[2] Mustafa K. Mehmet Ali and Faouzi Kamoun (1993), Neural networks for shortest path computation and routing in computer networks, *IEEE Trans. on Neural Networks*, *4*(6).

[3] G. W. Wilsonand G. S. Pawley(1988), On the stability of the travelling salesman poblem algorithm of Hopfield and tank, *Biol. Cybern.*, *58*, 63-70.

[4] J. J. Hopfield and D. W. Tank(1986), Neural computations of decisions in optimization problems, *Bol. Cybern*, *52*, 141-152.

[5] J. J. Hopfield(1982), Neural networks and physical systems with emergent collective computational abilities, *Proc. Nat. Acad. Scvi.*, *79*, 2554-2558.

[6] Jian-Hua Li 外 2인(1989), Analysis and synthesis of a class of neural networks: Linear systems operation on a closed hypercube, *IEEE Trans. on*

Circuits and Systems, *36*(11), 1405-1422.

[7] S. V. B. Aiyer, M. Niranjan, and F. Fallside (1990), A theoretical investigation into the performance of the Hopfield model, *IEEE Trans. Neural Networks*, *1*, 204-215.

[8] J. Li, A. N. Michael, and W. Porod(1988), Qualitative analysis and synthesis of a class of neural network, *IEEE Trans. Circuits Syst.*, *35*, 976-986.

Performance analysis of linear pre-processing hopfield network

Young-Hoon Ko* · Soo-Jong Lee** · Heung-Sik Noh***

Abstract

Since Dr. John J. Hopfield has proposed the Hopfield network, it has been widely applied to the pattern recognition and the routing optimization. The method of Jian-Hua Li improved efficiency of Hopfield network which input pattern's weights are regenerated by SVD(singluar value decomposition).

This paper deals with Li's Hopfield Network by linear pre-processing. Linear pre-processing is used for increasing orthogonality of input pattern set. Two methods of pre-processing are used, Hadamard method and random method. In manner of success rate, radom method improves maximum 30 percent than the original and hadamard method improves maximum 15 percent. In manner of success time, random method decreases maximum 5 iterations and hadamard method decreases maximum 2.5 iterations.

Key words : Hopfield network, pattern matching, neural network

* Professor, Dept. of Computer Engineering, Hyeobseong University
** Professor, Dept. of Computer Engineering, Hyeobseong University
*** Professor, Dept. of Computer Engineering, Hyeobseong University