Ch06_인공 신경망

이 장에서 다룰 내용

- 01_뇌의 동작 원리
- 02_뉴런
- 03_퍼셉트론
- 04_다층 신경망
- 05_홉필드 신경망
- 06_양방향 연상 메모리
- 07_자기조직 신경망
- 08_요약

01_뇌의 동작 원리

❖ 기계 학습

- 컴퓨터가 경험, 예, 유추를 통해 학습할 수 있게 하는 적응 메커니즘
- 학습 능력은 시간이 흐르면서 지능형 시스템의 성능을 개선함
- 적응형 시스템의 기초를 형성
- 기계 학습의 접근법
  - 결정 트리 (Decision Tree)
  - 베이시안 네트워크 (Bayesian Network)
  - 로지스틱 리그레션 (Logistic Regression)
  - 서포트 벡터 머신즈 (Support Vector Machines)
  - 인공 신경망 (Artificial Neural Network)
  - 유전 알고리즘 (genetic algorithm)

01_뇌의 동작 원리

- 기계 학습의 예 - 체스 게임
  - 딥 블루와 체스 게임 선수의 체스 게임
  - 딥 블루 - IBM에서 만든 초당 2억 개의 포지션을 분석할 수 있는 컴퓨터

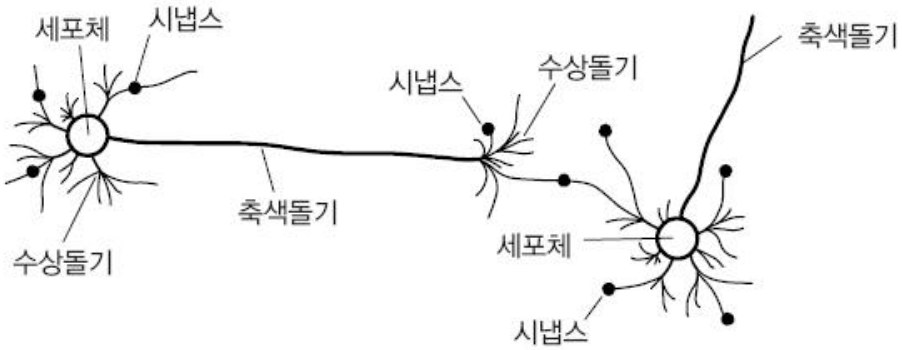
- 체스 게임을 통해 배운 것
  - 체스 프로그램은 경험을 통해 성능을 향상시켜야 한함
  - 기계는 반드시 학습 능력이 있어야 한다.

- 기계 학습의 다른 예 - Jeopardy 퀴즈 컨테스트
  - 왓슨 (Watson) 과 다른 두 명의 퀴즈 챔피언들 간의 대결에서 승리함

01_뇌의 동작 원리

❖ 인공 신경망

- 인공 신경망
  - 인간 뇌를 기반으로 한 추론 모델.
  - 인간 뇌의 추론 모델 - 뉴런(neuron) : [그림 6-1]



[그림 6-1] 생물학적인 신경망

  - 뉴런은 기본적인 정보처리 단위.
  - 인공 신경망의 주요 특징은 적응성을 가지고 있다는 것이다.

- 인간 뇌의 특징
  - 100 억개의 뉴런과 각 뉴런을 연결하는 6조 개의 시냅스의 결합체
  - 인간의 뇌는 현존하는 어떤 컴퓨터보다 빠르게 기능을 수행할 수 있음

01_뇌의 동작 원리

- 인간 뇌의 특징
  - 인간의 뇌는 매우 복잡하고, 비선형적이며, 병렬적인 정보 처리 시스템으로 생각할 수 있다.
  - 정보는 신경망 전체에 동시에 저장되고 처리된다.
  - 적응성에 따라 '잘못된 답'으로 이끄는 뉴런들 사이의 연결은 약화되고, '올바른 답'으로 이끄는 연결은 강화된다.

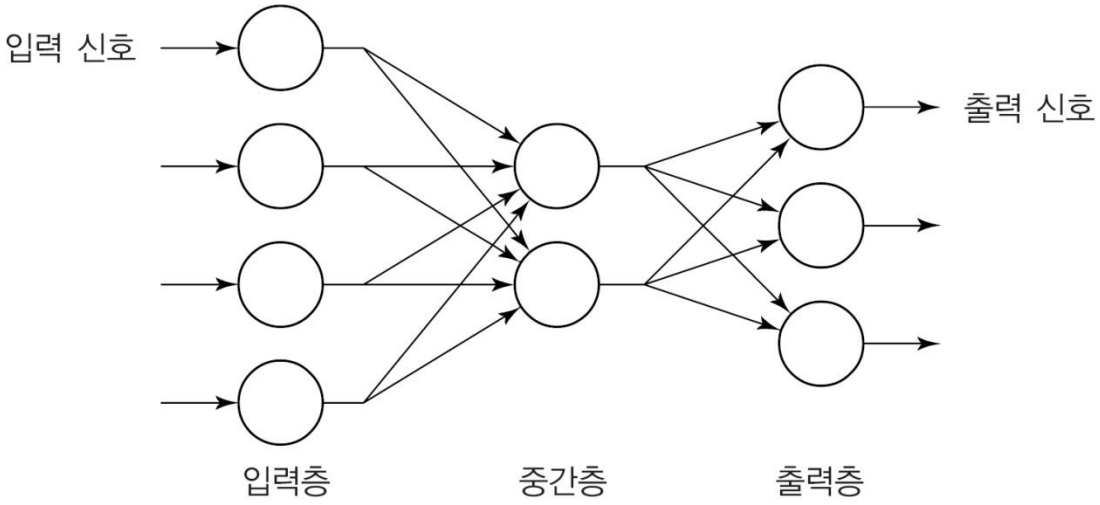
- 인공 신경망의 특징
  - 인간 뇌를 기반으로 모델링 함.
  - 인간 뇌의 적응성을 활용하여 '학습 능력'을 구현함.
  - 인공 신경망은 아직 인간의 뇌를 흉내내기에 미흡하다.

- 인공 신경망을 이용한 뇌 모델링
- 인공 신경망의 학습
- 인공 신경망의 가중치 조정

01_뇌의 동작 원리

❖ 인공 신경망 모델링

- 인간의 뇌 모델링
  - 생물학적인 뇌의 뉴런과 비슷하게 모델링함.
  - 인공 시경망은 뉴런이라는 아주 단순하지만 내부적으로 매우 복합하게 연결된 프로세스들로 이루어져 있음.
  - 뉴런은 가중치 있는 링크들로 연결되어 있음.
  - 각각의 뉴런은 연결을 통해 여러 입력 신호를 받지만 출력 신호는 오직 하나만 만듦.
  - 인공 신경망 구조 [그림 6-2]



[그림 6-2] 전형적인 인공 신경망의 구조

❖ 인공 신경망 모델링

- 인간의 뇌 모델링
  - 생물학적인 신경망과 인공 신경망의 유사점

[표 6-1] 생물학적인 신경망과 인공 신경망 사이의 유사점

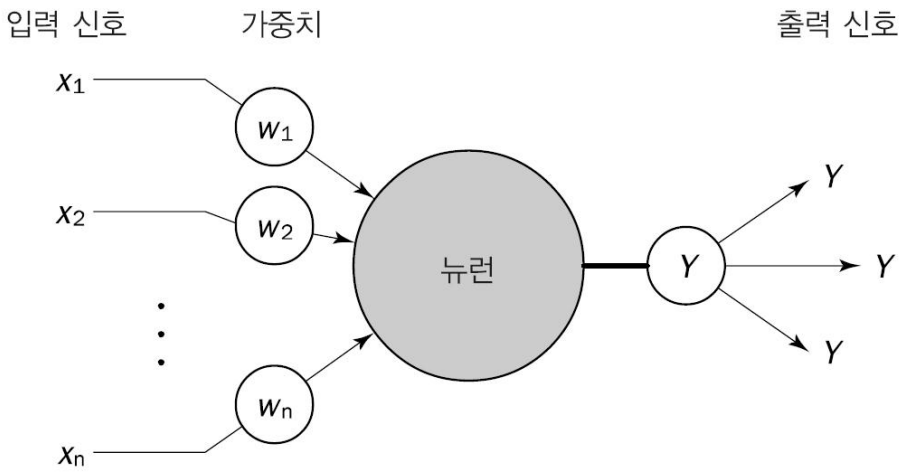
생물학적인 신경망	인공 신경망
세포체	뉴런
수상돌기	입력
축색돌기	출력
시냅스	가중치

- 인공 신경망의 학습
  - 신경망은 가중치를 반복적으로 조정하여 학습함.
  - 뉴런은 링크(link)로 연결되어 있고, 각 링크에는 그와 연관된 수치적인 가중치가 있다.
  - 가중치는 장기 기억을 위한 기본적인 수단으로,각 뉴런 입력 강도, 즉 중요도를 표현.

- 인공 신경망의 가중치 조정
  - 신경망의 가중치를 초기화하고 훈련 예제들의 집합에서 해당 가중치를 갱신.
  - 신경망의 구조를 먼저 선택하고, 어떤 학습 알고리즘을 사용할 것 인지 결정한 후, 신경망을 훈련시킨다.

02_단순한 계산 요소로서의 뉴런

- 뉴런의 특징
  - 입력 링크에서 여러 신호를 받아서 새로운 활성화 수준을 계산하고, 출력 링크로 출력 신호를 내보낸다.
  - 입력 신호는 미가공 데이터 또는 다른 뉴런의 출력이 될 수 있다.
  - 출력 신호는 문제의 최종적인 해(solution)거나 다른 뉴런에 입력될 수 있다.
  - 뉴런의 예 : [그림 6-3]



[그림 6-3] 뉴런의 도식

02_단순한 계산 요소로서의 뉴런

❖ 뉴런의 계산

▪ 뉴런의 출력 결정

- 워렌 맥클록(Warren McCulloch)과 월터 피츠(Walter Pitts)가 제안(1943년).
- 뉴런은 다음과 같은 전이 함수, 즉 활성화 함수(activation function)를 사용
- 활성화 함수를 이용한 출력 결정 순서

  1. 뉴런은 입력 신호의 가중치 합을 계산하여 임계값 θ와 비교한다.
  2. 가중치 합이 임계값보다 작으면 뉴런의 출력은 '-1'이다.
  3. 중치 합이 임계값과 같거나 크면 뉴런은 활성화되고, 뉴런의 출력은 '+1'이 된다

- 활성화 함수 : 식(6.1)

$$X = \sum_{i=1}^{n} x_i w_i \qquad (6.1)$$

$$Y = \begin{cases} +1 & \text{if } X \geq \theta \\ -1 & \text{if } X < \theta \end{cases}$$

X는 뉴런으로 들어가는 입력의 순 가중합

x_i는 입력 i의 값, w_i 는 입력i의 가중치, n은 뉴런의 입력 개수, Y는 뉴런의 출력

02_단순한 계산 요소로서의 뉴런

❖ 뉴런의 계산

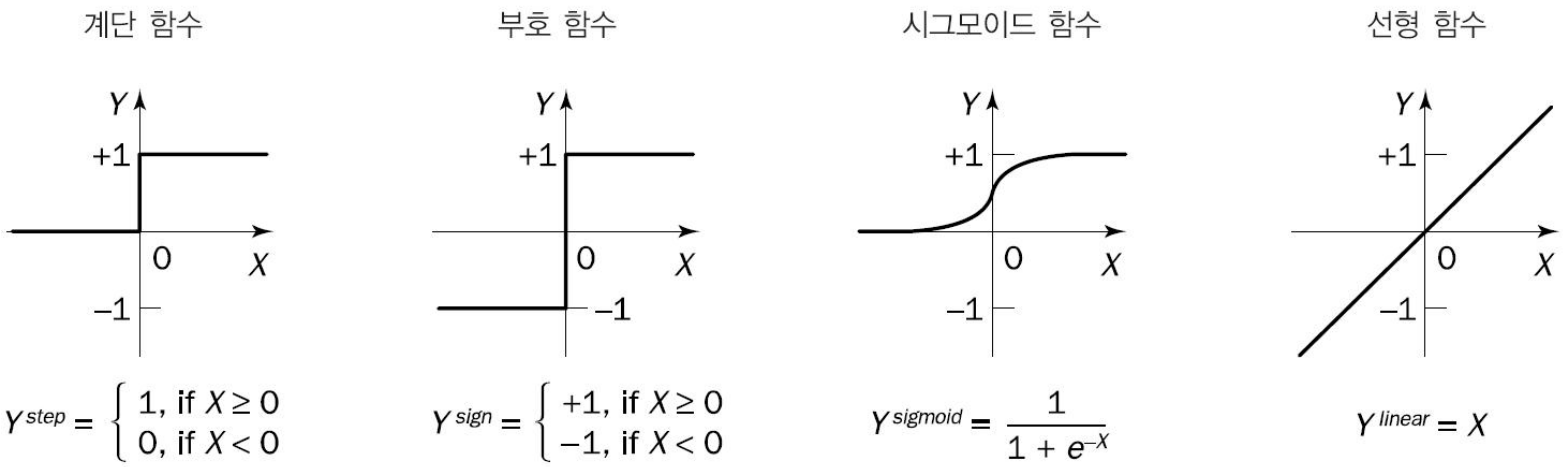
▪ 뉴런의 출력 결정

- 식(6.1)와 같은 활성화 함수를 부호함수라고 한다.
- 부호 활성화 함수를 사용하는 뉴런의 실제 출력 : 식(6.2)

$$Y = sign\left[\sum_{i=1}^{n} x_i w_i - \theta\right] \qquad (6.2)$$

▪ 뉴런의 활성화 함수

- 가장 일반적인 활성화 함수로는 계단, 부호, 선형, 시그모이드 함수가 있다. : [그림 6-4]



[그림 6-4] 뉴런의 활성화 함수들

02_단순한 계산 요소로서의 뉴런

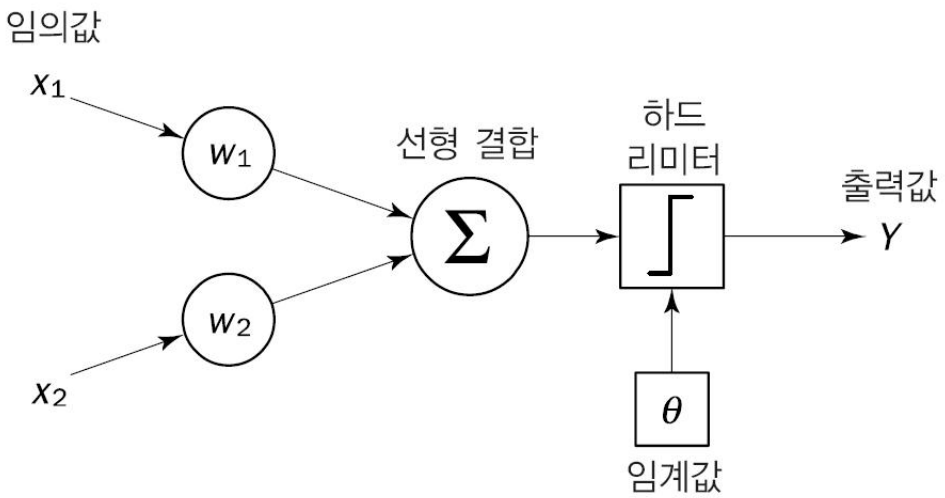
❖ 뉴런의 계산

- **뉴런의 활성화 함수**
  - 계단(step)과 부호(sign) 활성화 함수는 하드 리밋 함수(hard limit function)라고도 하며, 분류와 패턴인식 작업에서 결정을 내리는 뉴런에 주로 쓰인다.
  - 시그모이드 함수(sigmoid function)는 양과 음의 무한대 사이에 있는 입력값을 0~1 사이에 있는 적당한 값으로 바꾼다. 시그모이드 함수를 사용하는 뉴런은 역전파 신경망에 쓰인다.
  - 선형 활성화 함수(linear activation function)는 뉴런의 입력에 가중치가 적용된 것과 같은 값을 출력으로 내놓는다. 선형 함수를 사용하는 뉴런은 선형 근사에 주로 쓰인다.

- **단일 뉴런의 학습**
  - 프랭크 로젠블랫이 간단한 인공 신경망을 훈련시키기 위한 알고리즘인 퍼셉트론을 소개했다(1958년).
  - 퍼셉트론은 신경망의 가장 간단한 형태로 조정 가능한 시냅스 가중치와 하드 리미터(hard limiter)를 포함한 단일 뉴런으로 구성된다.

❖ 뉴런의 계산

- 단일 뉴런의 학습
  - 입력 노드가 두 개인 단층 퍼셉트론



[그림 6-5] 두 개의 입력 노드가 있는 단층 퍼셉트론

03_퍼셉트론

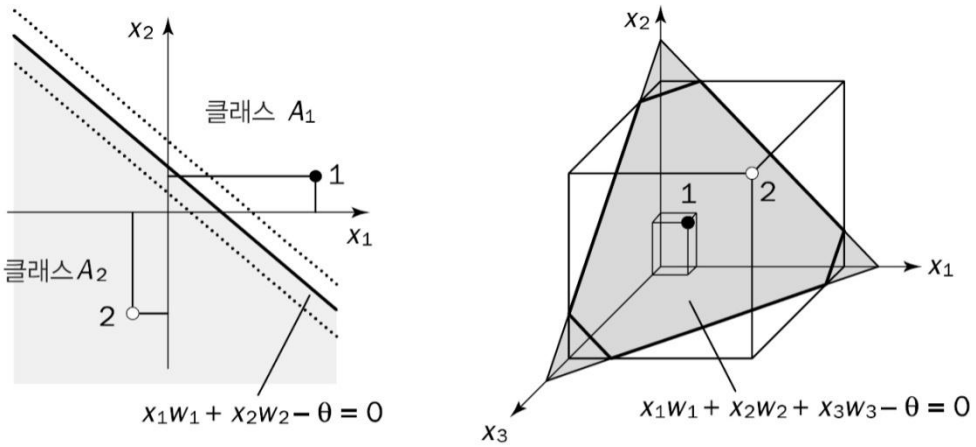
- 퍼셉트론의 특징
  - 로젠블랫 퍼셉트론의 동작 원리는 맥클록과 피츠의 뉴런 모델에 기반.
  - 퍼셉트론은 선형 결합기와 하드 리미터로 구성.
  - 입력의 가중합을 하드 리미터에 입력하고, 입력이 양이면 '+1', 음이면 '-1'을 출력.
  - 기본적인 퍼셉트론의 경우, 초평면(hyperplane)으로 n차원 공간을 두 개의 결정 영역으로 나뉜다.
  - 초평면을 선형 분리 함수로 정의한다.
  - 선형 분리 함수 : 식(6.3)

$$\sum_{i=1}^{n} x_i w_i - \theta = 0 \qquad (6.3)$$

  - 입력이 x_1과 x_2로 두 개인 경우, 결정 경계는 [그림 6-6]의 (a)에 보이듯 굵은 직선 형태로 나타낸다.
  - 경계선 오른편에 있는 점(1)은 클래스A_1에 속하고, 경계선 왼편에 있는 점(2)는 클래스 A_2에 속한다.
  - 임계값 θ는 결정 경계를 옮기는 데 쓰인다.

03_퍼셉트론

- **퍼셉트론의 특징**
  - **입력이 세 개일 때는 도식화 할 수 있다. [그림 6-6]의 (b)**
  - **[그림 6-6]은 입력이 세 개인 퍼셉트론에 대한 3차원 공간을 보여준다.**



(a) 입력이 두 개인 퍼셉트론 (b) 입력이 세 개인 퍼셉트론

[그림 6-6] 퍼셉트론에서의 선형 분리성

- **분할 식**

입력이 두 개인 퍼셉트론 $x_1w_1 + x_2w_2 - \theta = 0$

입력이 세 개인 퍼셉트론 $x_1w_1 + x_2w_2 + x_3w_3 - \theta = 0$

❖ 퍼셉트론 기본 학습

- 퍼셉트론 기본 학습 방법
  - 가중치를 조절하여 실제 출력과 목표 출력 간의 차이를 줄인다.
  - 보통 [-0.5, 0.5] 범위에서 초기 가중치를 임의로 할당한 후 훈련 예제와 일치하는 출력을 얻도록 갱신한다.
  - 오차 계산 식: 식(6.4)

$$e(p) = Y_d(p) - Y(p) \qquad p = 1, 2, 3, \ldots \qquad (6.4)$$

p번째 반복(퍼셉트론에 주어진 p번째 훈련 예제), 실제 출력이 Y(p)이고, 목표 출력 $Y_d(p)$

  - 오차 e(p)가 양이면 퍼셉트론의 출력 Y(p)를 증가시켜야 하고, 그 값이 음이면 Y(p)를 감소시킨다.

- 퍼셉트론 학습 규칙
  - 로젠블랫이 제안(1960년대)
  - 오차 e(p)를 이용함 :

    -. 오차 e(p)가 양이면 퍼셉트론의 출력 Y(p)를 증가한다.

    -. 오차 e(p)가 음이면 Y(p)를 감소한다.

    -. 각 퍼셉트론 입력에 대해 x(p) × w(p) 값이 총합 X(p)에 기여한다는 것을 고려하면, x(p)가 양일 때 가중치 w(p)가 커지고Y(p)도 커진다.

    -. x(p)가 음일 때 가중치 w(p)가 커지면 Y(p)가 줄어든다.

03_퍼셉트론

- 퍼셉트론 학습 규칙
  - 퍼셉트론 학습 규칙 식

$$w_i(p+1) = w_i(p) + \alpha \times x_i(p) \times e(p) \qquad (6.5)$$

α는 학습률(learning rate)로, 1보다 작은 양의 상수.

- 분류 작업을 위한 퍼셉트론 훈련 알고리즘 : 4단계 알고리즘
  - 1단계: 초기화
    초기 가중치 $w_1, w_2, \ldots, w_n$과 임계값 θ를 [-0.5, 0.5] 구간의 임의의 값으로 설정.
  - 2단계: 활성화
    입력 $x_1(p), x_2(p), \ldots, x_n(p)$와 목표 출력 $Y_d(p)$를 적용하여 퍼셉트론을 활성화.
    반복 횟수 p=1에서 실제 출력을 계산.

$$Y(p) = step\left[\sum_{i=1}^{n} x_i(p) w_i(p) - \theta\right] \qquad (6.6)$$

n은 퍼셉트론의 입력 개수고, step은 계단 활성화함수(activation function)

03_퍼셉트론

- 분류 작업을 위한 퍼셉트론 훈련 알고리즘 : 4단계 알고리즘
  - 3단계: 가중치 학습
    퍼셉트론의 가중치를 갱신한다.

$$w_i(p+1) = w_i(p) + \Delta w_i(p) \qquad (6.7)$$

    $\Delta w_i(p)$는 p번째 반복했을 때의 가중치 보정값. 가중치 보정값은 델타 규칙(delta rule)으로 계산한다.

$$\Delta w_i(p) = \alpha \times x_i(p) \times e(p) \qquad (6.8)$$

  - 4단계: 반복
    반복 횟수 p값을 1 증가시키고, 2단계로 돌아가서 수렴할 때까지 과정을 반복한다.

❖ 퍼셉트론 학습 방법

- **기본적인 논리 연산자 학습**
  - AND, OR, Exclusive-OR와 같은 기본적인 논리 연산자의 기능을 수행하도록 훈련.
  - 연산자 AND, OR, Exclusive-OR의 진리표: [표 6-2]

[표 6-2] 기본적인 논리 연산자들의 진리표

입력 변수		AND	OR	Exclusive-OR
x_1	x_2	$x_1 \cap x_2$	$x_1 \cup x_2$	$x_1 \oplus x_2$
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

- **AND와 OR 연산자 학습**
  - 학습 순서
    1. 하나의 에폭 나타내는 네 개의 연속된 입력 패턴으로 퍼셉트론을 활성화한다.
    2. 퍼셉트론의 가중치는 각각의 활성화 이후에 갱신된다
    3. 가중치가 일관된 수치 집합으로 수렴할 때까지 이 과정을 반복한다

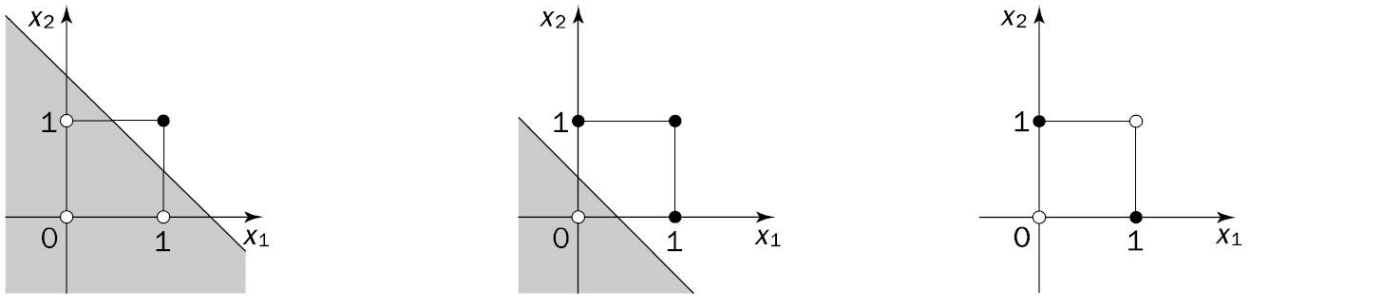
- AND와 OR 연산자 학습
  - AND 논리 연산자의 퍼셉트론 학습 결과 : [표 6-3]

[표 6-3] 퍼셉트론 학습의 예: AND 논리 연산자 (임계값: $\theta = 0.2$, 학습률: $\alpha = 0.1$)

	입력		목표 출력	초기 가중치		실제 출력	오차	최종 가중치	
에폭	x_1	x_2	Y_d	w_1	w_2	Y	e	w_1	w_2
1	0	0	0	0.3	−0.1	0	0	0.3	−0.1
	0	1	0	0.3	−0.1	0	0	0.3	−0.1
	1	0	0	0.3	−0.1	1	−1	0.2	−0.1
	1	1	1	0.2	−0.1	0	1	0.3	0.0
2	0	0	0	0.3	0.0	0	0	0.3	0.0
	0	1	0	0.3	0.0	0	0	0.3	0.0
	1	0	0	0.3	0.0	1	−1	0.2	0.0
	1	1	1	0.2	0.0	1	0	0.2	0.0
3	0	0	0	0.2	0.0	0	0	0.2	0.0
	0	1	0	0.2	0.0	0	0	0.2	0.0
	1	0	0	0.2	0.0	1	−1	0.1	0.0
	1	1	1	0.1	0.0	0	1	0.2	0.1
4	0	0	0	0.2	0.1	0	0	0.2	0.1
	0	1	0	0.2	0.1	0	0	0.2	0.1
	1	0	0	0.2	0.1	1	−1	0.1	0.1
	1	1	1	0.1	0.1	1	0	0.1	0.1
5	0	0	0	0.1	0.1	0	0	0.1	0.1
	0	1	0	0.1	0.1	0	0	0.1	0.1
	1	0	0	0.1	0.1	0	0	0.1	0.1
	1	1	1	0.1	0.1	1	0	0.1	0.1

03_요약

- Exclusive-OR 연산자 학습
  - 단층 퍼셉트론을 학습할 수 없다.
  - 선행 분리가 불가능하기 때문이다.
  - AND, OR, Exclusive-OR 함수의 2차원 도면 : [그림 6-7]
    함수의 출력이1인입력공 간의 점은 검은색으로 출력이 0인 점은 흰색.



(a) AND 연산 ($x_1 \cap x_2$) (b) OR 연산 ($x_1 \cup x_2$) (c) Exclusive-OR 연산 ($x_1 \oplus x_2$)

[그림 6-7] 기본적인 논리 연산자들의 2차원 도면

  - [그림 6-7]를 통해 다음을 알 수 있다.
    1. (a)와 (b)에서는 검은 점과 흰 점을 구분하여 직선을 그릴 수 있지만, (c)의 점들은 직선으로 분리할 수 없다.
    2. 퍼셉트론은 모든 검은 점과 모든 흰 점을 분리하는 직선이 있을 때만 함수로 표현할 수 있다.

03_요약

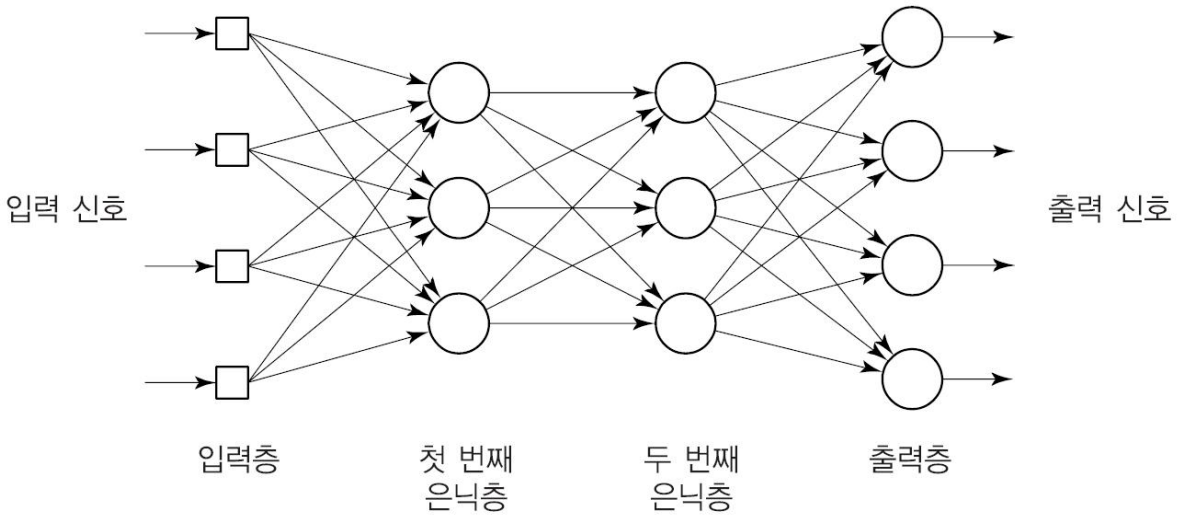
- 단층 퍼셉트론 학습의 한계
  - 단층 퍼셉트론은 선형 분리 가능한 함수만 학습할 수 있다.
  - 선행 분리 가능한 함수들이 많지 않다는 점을 고려해 볼 때 좋지 않다.
  - 단층 퍼셉트론의 계산적 한계는 민스키와 페퍼트가 저술한『Perceptrons』(MIT Press)에서 수학적으로 분석하여 로젠블랫 퍼셉트론이 전체적인 일반화를 할 수 없다는 것을 증명.

- 단층 퍼셉트론 학습의 한계 극복
  - 다층 신경망으로 단층 퍼셉트론을 극복할 수 있다.
  - 역전파 알고리즘으로 학습한 다층 퍼셉트론처럼 발전된 형태의 신경망을 사용하면 로젠블랫 퍼셉트론의 한계를 극복할 수 있다는 것이 증명되었다.

❖ 다층 신경망

▪ 다층 신경망의 구조

- 하나 혹은 그 이상의 은닉층이 있는 피드포워드 신경망(feedforward neural network).
- 일반적으로 이 신경망은 공급 뉴런으로 이루어진 입력층(input layer) 하나, 계산 뉴런들로 이루어진 하나 이상의 은닉층(hidden layer), 계산 뉴런들로 이루어진 출력층(output layer) 하나로 이루어져 있음.
- 입력 신호는 한 층씩 순방향으로 전파됨.
- 다층 신경망의 예 : [그림 6-8] 두 개의 은닉층이 있는 다층 신경망.



[그림 6-8] 두 개의 은닉층이 있는 다층 신경망

❖ 다층 신경망

- 다층 신경망의 구조
  - 다층 신경망의 층에는 입력층, 출력층, 은닉층이 있다.
  - 각 층에는 각각 자신만의 특정 함수가 있다.

- 입력층
  - 외부에서 받아들인 입력 신호를 은닉층의 모든 뉴런으로 보낸다.
  - 계산을 위한 뉴런은 거의 들어 있지 않다.

- 출력층
  - 은닉층에서 출력 신호, 즉 자극 패턴을 받아 들이고 전체 신경망의 출력 패턴을 정한다.

- 은닉층
  - 입력의 특성을 파악한다. 뉴런의 가중치는 입력 패턴에 숨겨져 있는 특성을 나타낸다.
  - 출력층이 출력 패턴을 정할 때, 이 특성을 사용한다.
  - 은닉층은 목표 출력을 '숨기고' 있다. 즉, 은닉층의 목표 출력은 해당 층에서 자체적으로 결정된다.
  - 신경망에 은닉층이 두 개 이상 들어갈 수 있다.

03_요약

❖ 다층 신경망의 학습

- 다층 신경망의 학습은 퍼셉트론과 유사하게 진행된다.
- 신경망은 출력 패턴을 계산하고 오차가 있다면, 즉 실제와 목표 출력 간에 차이가 있다면 이 오차를 줄이도록 가중치를 조절한다.
- 다층 신경망에서는 가중치가 여러 개인데, 각각의 가중치는 두 개 이상의 출력에 영향을 미친다.

- **역전파 신경망**
  - 세 개 또는 네 개의 층이 있는 다층망이다.
  - 다층 신경망의 오차 원인을 평가한다.
  - 영향을 미치는 여러 가중치들 사이에서 오차의 원인을 정하고 나누는 데 사용된다.

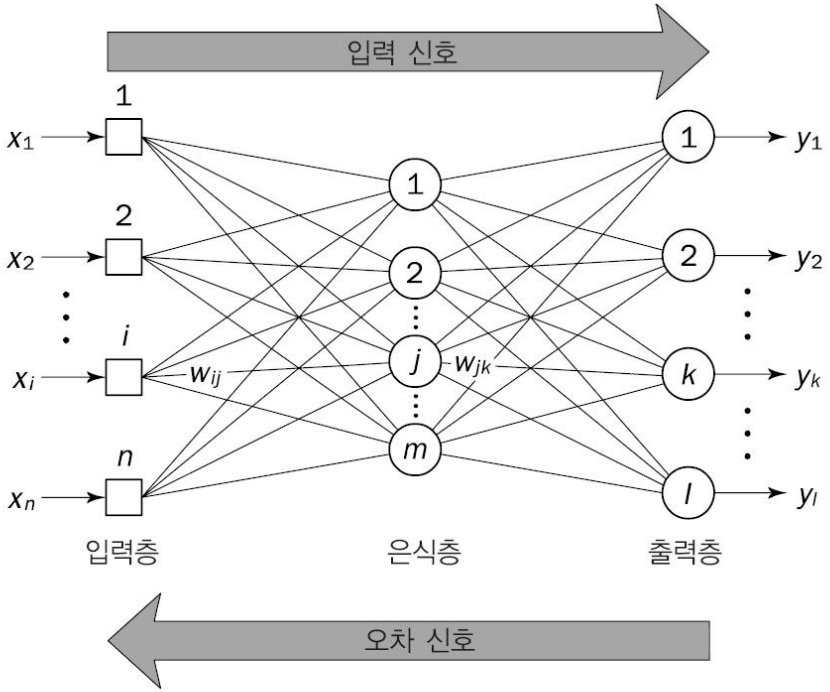
- **역전파 신경망의 학습 알고리즘**
  - 학습 알고리즘의 두 단계.

    1. 훈련 입력 패턴을 신경망의 입력층에 전달
    2. 신경망은 출력층에서 출력 패턴이 생성될 때까지 층에서 층으로 입력 패턴을 전파

  - 출력 패턴이 목표 패턴과 다르면 그 오차를 계산한 후 출력층에서 입력층까지 신경망을 따라 거꾸로 전파한다.
  - 오차가 전파되면서 가중치가 수정된다.

❖ 역전파 학습 법칙

- 역전파 망 구조
  - 층이 세 개인 신경망 [그림 6-9]
  - 인덱스 i, j, k는 각각 입력층, 은닉층, 출력층 뉴런.



[그림 6-9] 세 개의 층이 있는 역전파 신경망

입력층과 은식층 사이에 있는 가중치 16개 은식충에서 출력층 16개
은식충 8개 -> 40개의 parameter

오차신호는 역으로 전달된다.

04_다층 신경망

- **역전파 망 구조 : [그림 6-9]**
  - **오차 신호는 은닉층으로 거꾸로 전파된다. p번째 반복에서 뉴런 k의 출력에 대한 오차 신호는 식(6.10)과 같이 정의한다.**

$$e_k(p) = y_{d,k}(p) - y_k(p) \qquad (6.10)$$

  $y_{d,k}(p)$는 p번째 반복에서 뉴런 k의 목표 출력

  - **출력층에 있는 뉴런 k는 자신의 목표 출력을 알고 있으므로 가중치 w_{jk}를 바로 갱신할 수 있다. 출력층의 가중치를 갱신하는 규칙은 (6.7)의 퍼셉트론 학습 규칙과 비슷하다.**

$$w_{jk}(p+1) = w_{jk}(p) + \Delta w_{jk}(p) \qquad (6.11)$$

  $w_{jk}(p)$는 가중치 보정값이다.

  - **입력 신호 x_i를 사용할 수 없다면 은닉층의 뉴런 j의 출력 y_i를 사용한다.**

$$\Delta w_{jk}(p) = \alpha \times y_j(p) \times \delta_k(p) \qquad (6.12)$$

  $\delta_k(p)$는 p번째 반복에서 출력층에 있는 뉴런 k에 대한 오차 기울 기다.

- **오차 기울기**(error gradient)
  - 오차 기울기는 활성화 함수의 미분에 뉴런의 출력 오차를 곱한 것이다.
  - 출력층의 뉴런 k에 대해 식(6.13)을 얻는다.

$$\delta_k(p) = \frac{\partial y_k(p)}{\partial X_k(p)} \times e_k(p) \qquad (6.13)$$

Sigmoid 미분값

$y_k(p)$는 p번째 반복에서 뉴런 k의 출력이고, $X_k(p)$는 같은 반복에서 뉴런 k로 들어가는 순 가중 입력이다.

  - 시그모이드 활성화 함수 : 식(6.14)

Sigmoid 미분값

$$\delta_k(p) = y_k(p) \times [1 - y_k(p)] \times e_k(p) \qquad (6.14)$$

여기서 $y_k(p)$는 다음과 같다.

$$y_k(p) = \frac{1}{1 + \exp[-X_k(p)]}$$

X = 그냥 출력값

❖ 은닉층 뉴런에 대한 가중치 보정값

- 은닉층의 가중치 보정
  - 출력층과 같은 식을 적용할 수 있다. : 식(6.15)

$$\triangle w_{ij}(p) = \alpha \times x_i(p) \times \delta_i(p) \qquad (6.15)$$

$\delta_j(p)$는 은닉층에 있는 뉴런 j에서의 오차 기울기를 나타낸다.

$$\delta_j(p) = y_j(p) \times [1 - y_j(p)] \times \sum_{k=1}^{l} \delta_k(p) w_{jk}(p)$$

l은 출력층에 있는 뉴런의 개수다. $y_j(p)$와 $X_j(p)$는 다음과 같다.

$$y_j(p) = \frac{1}{1 + e^{-X_j(p)}}$$

$$X_j(p) = \sum_{i=1}^{n} x_i(p) \times w_{ij}(p) - \theta_j$$

n은 입력층에 있는 뉴런의 수 이다.

❖ 은닉층 뉴런에 대한 가중치 보정값

- 역전파 학습 알고리즘
  - 식(6.15)를 이용하여 역전파 학습 알고리즘을 이끌어 낸다.
  - 총 4단계로 구성

- 1단계 : 초기화
  - 가중치 초기화는 각 뉴런별로 이루어 진다.
  - 신경망의 모든 가중치와 임계값의 수준을 좁은 범위 안에서 균등 분포를 따라 임의의 수로 놓는다.

$$\left(-\frac{2.4}{F_i}, \quad +\frac{2.4}{F_i}\right)$$

F_i는 신경망에 있는 뉴런 i의 총 입력 개수다.

- 2단계 : 활성화
  - 입력과 목표 출력을 적용하여 역전파 신경망을 활성화한다.
  - 은닉층에 있는 뉴런의 실제 출력을 계산한다.

$$y_j(p) = sigmoid\left[\sum_{i=1}^{n} x_i(p) \times w_{ij}(p) - \theta_j\right]$$

04_다층 신경망

- 2단계 : 활성화
  - 출력층에 있는 뉴런의 실제 출력을 계산한다.

$$y_k(p) = sigmoid\left[\sum_{j=1}^{m} x_{jk}(p) \times w_{jk}(p) - \theta_k\right]$$

sigmoid는 시그모이드 활성화 함수, m은 출력층에 있는 뉴런 k의 입력 개 수

- 3단계 : 가중치 학습
  - 출력 뉴런과 연관된 오차를 역방향으로 전파시키면서 역전파 신경망의 가중치를 갱신해 나간다.
  - 출력층에 있는 뉴런에 대해 오차 기울기를 계산한다.

$$\delta_k(p) = y_k(p) \times [1 - y_k(p)] \times e_k(p)$$

뉴런의 출력 가중치 갱신 $w_{jk}(p+1) = w_{jk}(p) + \Delta w_{jk}(p)$

  - 은닉층에 있는 뉴런의 오차 기울기를 계산한다.

$$\delta_j(p) = y_j(p) \times [1 - y_j(p)] \times \sum_{k=1}^{l} \delta_k(p) \times w_{jk}(p)$$

은닉층에서의 가중치를 갱신 $w_{ij}(p+1) = w_{ij}(p) + \Delta w_{ij}(p)$

04_다층 신경망

- 2단계 : 활성화
  - 출력층에 있는 뉴런의 실제 출력을 계산한다.

$$y_k(p) = sigmoid\left[\sum_{j=1}^{m} x_{jk}(p) \times w_{jk}(p) - \theta_k\right]$$

sigmoid는 시그모이드 활성화 함수, m은 출력층에 있는 뉴런 k의 입력 개 수

- 3단계 : 가중치 학습
  - 출력 뉴런과 연관된 오차를 역방향으로 전파시키면서 역전파 신경망의 가중치를 갱신해 나간다.
  - 출력층에 있는 뉴런에 대해 오차 기울기를 계산한 후 뉴런의 출력 가중치 갱신한다.

$$w_{jk}(p+1) = w_{jk}(p) + \Delta w_{jk}(p)$$

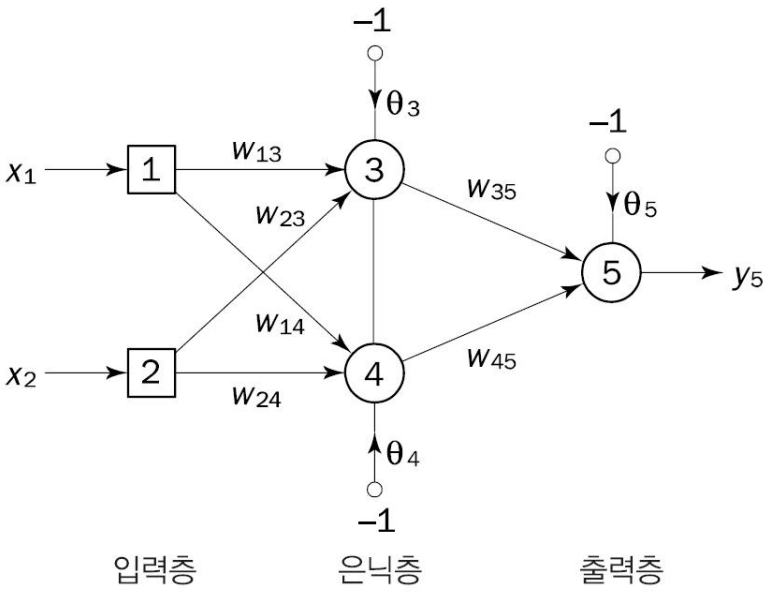
  - 은닉층에 있는 뉴런의 오차 기울기를 계산한 후 은닉층의 가중치를 갱신한다

$$w_{ij}(p+1) = w_{ij}(p) + \Delta w_{ij}(p)$$

- 4단계 : 반복
  - 반복 횟수 p값을 1 증가시키고, 2단계로 돌아가서 선택한 오차 기준을 만족할 때까지 과정을 반복한다.

04_다층 신경망

- Exclusive-OR 연산을 수행하기 위한 신경망
  - 층이 세 개인 역전파 신경망 : [그림 6-10]
  - Exclusive-OR 연산을 수행

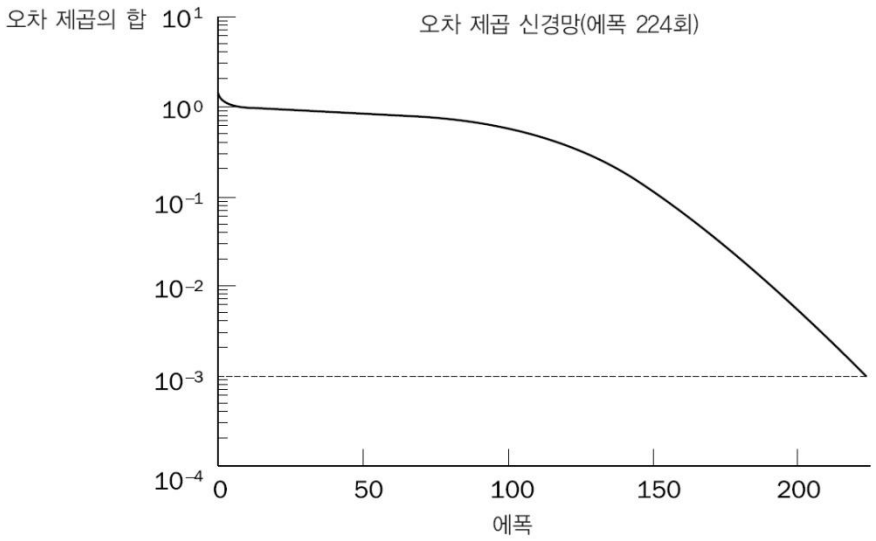


[그림 6-10] Exclusive-OR 연산을 수행하는 층이 세 개인 신경망

- 오차 제곱의 합(sum of the squared errors)
  - 신경망의 성능을 보여주는 유용한 지표다.
  - 한 패스 동안의 모든 훈련 집합 또는 에폭에 대한 오차 제곱의 합이 충분히 작으면 신경망이 수렴했다고 생각할 수 있다.

04_다층 신경망

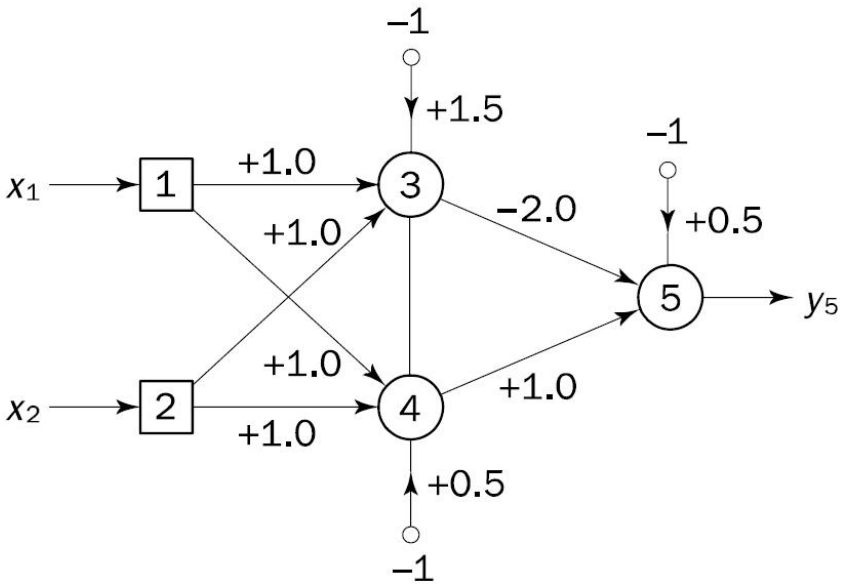
- **오차 제곱의 합**(sum of the squared errors)
  - **충분히 작은 오차 제곱의 합이** 0.001**일 때, 학습곡선** : [**그림** 6-11]



[그림 6-11] Exclusive-OR 연산자의 학습 곡선

- 학습에 사용된 에폭들의 횟수에 대해 오차 제곱의 합.
- 학습 곡선은 신경망이 얼마나 빨리 학습하고 있는지를 보여줌.

- **다층망이 형성하는 결정 경계**
  - Exclusive-OR **연산을 풀기 위한 신경망** : [**그림** 6-12]

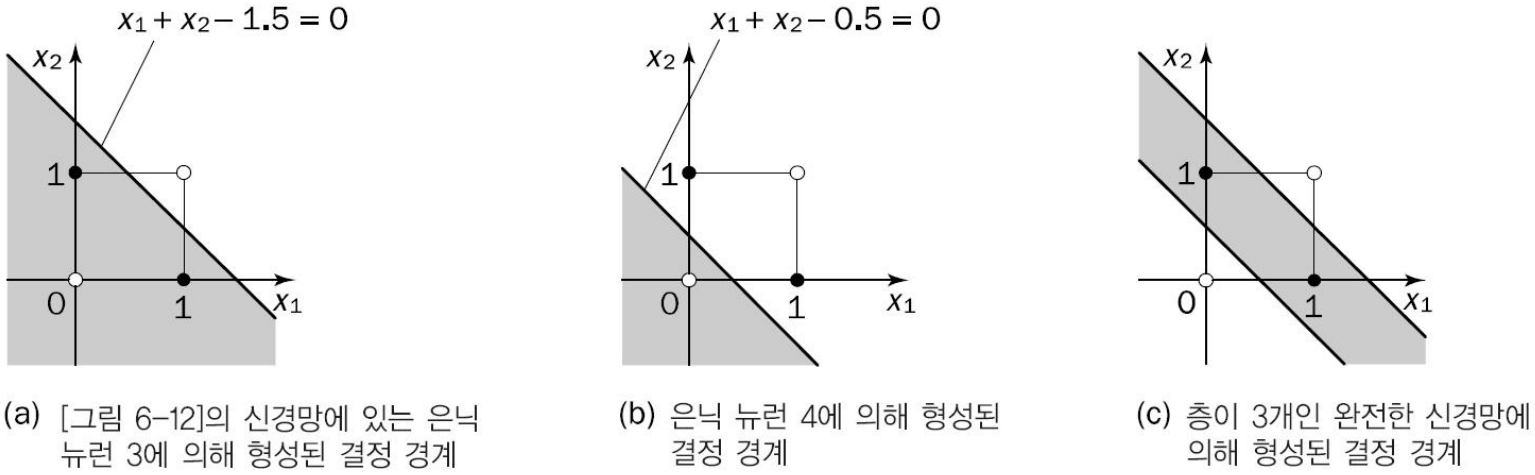


[그림 6-12] 맥클록-피츠 모델에 의한 Exclusive-OR 연산을 풀기 위한 신경망

- **부호함수를 사용하는 맥클록과 피츠의 모델로 은닉층과 출력층에 있는 뉴런을 표현.**
- **시그모이드 활성화 함수를 사용하는 뉴런들이 만들어낸 결정 경계를 그리는 일은 조금 어려울 수 있다.**
- **[그림 6-12]는 신경망 또는 Exclusive-OR 연산을 수행하도록 훈련 받은 것이다.**

04_다층 신경망

- Exclusive-OR 연산을 수행하기 위한 신경망
  - 신경망을 검은 점과 흰점으로 분리하여 Exclusive-OR 문제를 해결 : [그림 6-13]



(a) [그림 6-12]의 신경망에 있는 은닉 뉴런 3에 의해 형성된 결정 경계

(b) 은닉 뉴런 4에 의해 형성된 결정 경계

(c) 층이 3개인 완전한 신경망에 의해 형성된 결정 경계

[그림 6-13] 은닉층에 있는 뉴런 3과 뉴런 4에 의해 형성된 결정 경계

- 기계 학습에 대한 역전파 학습의 적합성
  - 역전파 학습은 널리 쓰이고 있지만, 문제에 대한 면역성이 없다.
  - 역전파 학습은 생물학적인 영역에서 제대로 학습하지 못 한다.
  - 계산이 방대하기 때문에 학습이 느리다.

  → 역전파 학습을 인간의 뇌와 같은 학습 방법을 흉내 낸 과정이라 보기 어렵다.

  - 역전파 학습을 기계 학습에 사용하기는 적합하기 않다

05_다층 신경망에서의 가속 학습

❖ 가속 학습

- 역전파 알고리즘의 계산 효율을 높이는 방법

- 일반적인 다층 신경망의 학습
  - 시그모이드 활성화 함수가 쌍곡 탄젠트로 표현될 때 조금 더 빠르게 학습한다.

$$Y^{tanh} = \frac{2a}{1 + e^{-bX}} - a \qquad (6.16)$$

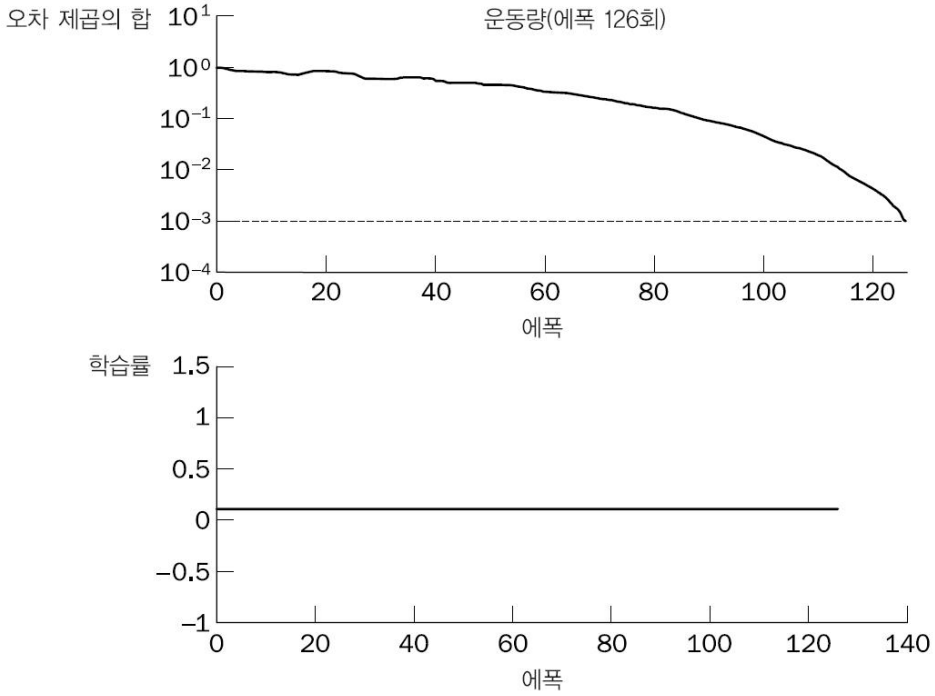
  - a와 b는 상수로 a = 1.716, b = 0.667가 적당하다(Guyon, 1991)
  - 델타 큐칙에 운동량 항을 포함시킴으로써 학습을 더 빠르게 할 수 있다.
  - 일반화된 델타 큐칙 : 식(6.17)

$$\Delta w_{jk}(p) = \beta \times \Delta w_{jk}(p-1) + \alpha \times y_j(p) \times \delta_k(p) \qquad (6.17)$$

- 운동량 상수의 필요성
  - 와트라우스와 제이콥스가 관찰.
  - 역전파 알고리즘에 운동량을 포함시키면 학습하는 데 안정화 효과를 가져온다.

05_다층 신경망에서의 가속 학습

- 운동량 상수의 필요성
  - 운동량을 포함시키면 내리막 방향일 때는 하강하는 데 가속이 붙고 학습 표면이 봉우리와 골짜기 상태일 때는 학습 과정이 감속되는 경향을 보인다.
  - 운동량을 포함한 Exclusive-OR 연산 학습 결과 : [그림 6-14]



[그림 6-14] 운동량을 이용한 학습

순서 역전파 알고리즘과 비교해보면 에폭의 횟수가 224에서 126으로 줄어듬.

05_다층 신경망에서의 가속 학습

❖ 역전파 학습의 수렴 가속

- 훈련 중에 학습률 매개변수를 조절
  - 가장 효과적인 방법 중 하나.
  - 작은 학습률 매개변수 α는 한 반복에서 그 다음으로 신경망의 가중치에 작은 변화를 일으켜 매크러운 학습 곡선을 나타낸다.
  - 매개변수 α를 큰 값으로 설정하면 가중치가 크게 변하므로 불안정한 상태가 되며, 신경망이 변동할 수 있다.
  - 매개변수 α를 큰 값으로 설정하면 훈련 과정을 더 빠르게 수렴할 수 있다.

- 휴리스틱을 이용한 수렴 가속
  - 수렴을 가속시키면서도 불안정한 상태를 피할 수 있다.
  - 두 가지 휴리스틱을 적용할 수 있다.

  - 휴리스틱1:

    오차 제곱의 합의 변화량이 몇 번의 연속적인 에폭에 대해 계속 같은 부호로 나타나면 학습률 매개변수 α값을 증가시킨다.
  - 휴리스틱2:

    오차 제곱의 합의 변화량의 부호가 몇 번의 연속적인 에폭에 대해 계속 엇갈리면 학습률 매개변수α을 감소시킨다.

06_홉필드 신경망

- 홉필드 신경망은 존 홉필드가 제안한 순환 신경망이다.
- 순환 신경망은 출력에서부터 입력까지 피드백 루프(feedback loop)가 있다. 피드백 루프는 신경망의 학습 능력에 많은 영향을 미친다.

❖ 순환 신경망의 학습

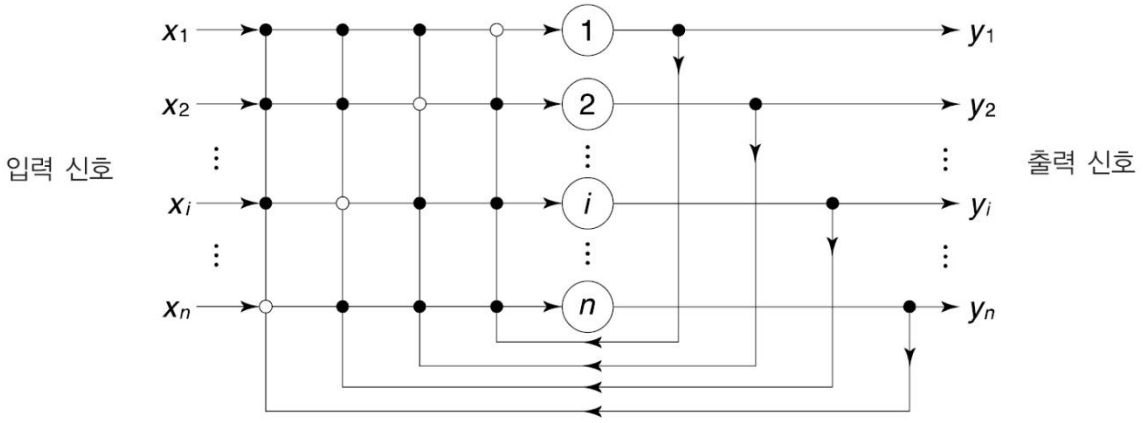
- 순환 신경망의 학습 방법
  - 새로운 입력을 적용한 후, 신경망의 출력을 계산하고 다시 거꾸로 보내 해당 입력값을 조정한다. 그런 후 출력을 다시 계산한다.
  - 출력이 상수값이 될 때까지 과정을 반복한다.

- 출력이 항상 상수가 되는가
  - 성공적으로 반복했다고 해서 출력값의 변화가 항상 작아지는 것은 아니며, 오히려 혼란스러운 동작을 보이기도 한다.

- 순환 신경망의 안전성
  - 1960년대~1970년대까지 신경망의 안정성을 예측하지 못했다.
  - 1982년 홉필드가 동적 안정 신경망에서 정보를 저장하는 것에 대한 물리적 원칙을 세운 이후에야 안전성 문제를 해결할 있었다.

06_홉필드 신경망

- 순환 신경망의 안전성
  - 단층 n-뉴런 홉필드 신경망



[그림 6-17] 단층 n-뉴런 홉필드 신경망

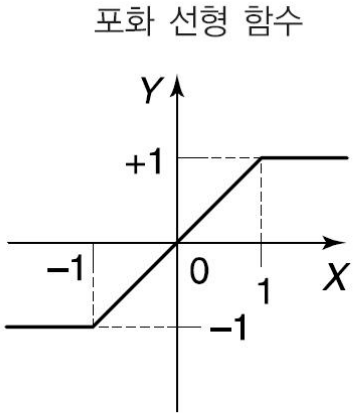
  - 홉필드가 제안한 홉필드 신경망은 계산 요소로서 부호 활성화 함수를 따른 맥클록-피츠 뉴런을 사용한다.

- 부호 활성화 함수
  - 부호 함수와 비슷하게 동작한다.
  - 뉴런의 가중 입력이 0 보다 작으면 출력은 -1 이고, 0 보다 크면 출력은 +1이다.
  - 0이면 출력은 바뀌지 않는다. 뉴런이 이전 출력이 +1이었든 -1이었든 상관 없이 이전 상태로 남는다.

- **부호 활성화 함수**
  - **포화 선형 함수** : **식**(6.18)

$$Y^{sign} = \begin{cases} +1 & \text{if } X > 0 \\ -1 & \text{if } X < 0 \\ Y & \text{if } X = 0 \end{cases} \qquad (6.18)$$

  - **부호 활성화 함수를 포화 선형 함수로 바꿀 수 있다**. [**그림** 6-18]



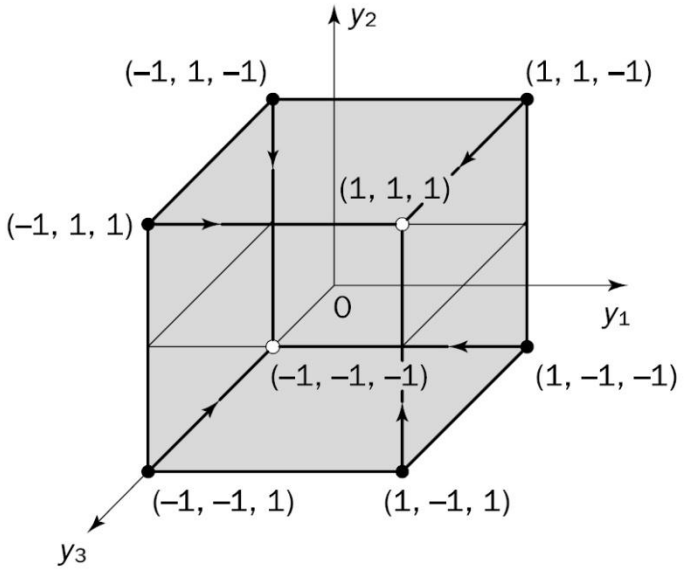
$$Y^{satlin} = \begin{cases} X & \text{if } -1 < X < 1 \\ +1 & \text{if } X \geq 1 \\ -1 & \text{if } X \leq -1 \end{cases}$$

[그림 6-18] 포화 선형 활성화 함수

함수는 [-1, 1] **범위에서 순수 선형 함수**(pure linear function)**처럼 동작하고, 범위를 벗어나면 부호 함수**(sign function)**처럼 동작한다**

❖ 홉필드 신경망의 동작

- 홉필드 신경망의 동작은 기하학적으로 나타낼 수 있음 : [그림 6-19]
  - 일반적으로 뉴런이 n개인 신경망에는 가능한 상태가 2^n개 있으며, 이는n차원 하이퍼큐브(hypercube)와 관련이 있다.



[그림 6-19] 3-뉴런 홉필드 신경망에서 가능한 상태를 보여주는 정육면체

  - 각 상태를 하나의 정점으로 표현한다.
  - 새로운 입력 벡터를 적용하면 신경망이 안정될 때까지 이쪽 정점에서 다른 쪽 정점으로 이동한다.

06_홉필드 신경망

- 안정적인 상태 정점의 결정
  - 가중치 행렬W, 현재 입력 벡터X, 임계값 행렬 θ가 안정적인 상태 정점을 결정한다.
  - 일부 입력벡터 중 잘못되었거나 불완전한 부분이 있다면, 몇 번의 반복 후에 안정적인 상태 정점으로 수렴할 것이다.

- 홉필드 신경망 검사
  - 입력 벡터X를 적용하여 신경망을 활성화한 후 실제 출력 벡터 Y를 계산한다.
  - 최초의 입력벡터 X와 결과를 비교한다.

$$\mathbf{Y}_m = sign\left(\mathbf{W}\ \mathbf{X}_m - \theta\right) \qquad m = 1, 2, \ldots, M \qquad (6.21)$$

θ는 임계값 행렬이다.

  - 홉필드 신경망은 오차 수정 신경망으로 작동한다.

❖ 홉필드 신경망 훈련 알고리즘

- 3단계로 구성 : 저장, 검사, 복구
- 1단계: 저장
  - n-뉴런 홉필드 신경망은M개의 기본 기억 $Y_1, Y_2, \ldots, Y_M$ 집합을 저장해야 한다.
  - 뉴런 i에서 뉴런 j로 가는 시냅스 가중치는 식(6.22)와 같이 계산한다.

$$w_{ij} = \begin{cases} \sum_{m=1}^{M} y_{m,i}\, y_{m,j} & i \neq j \\ 0 & i = j \end{cases} \qquad (6.22)$$

여기서 $y_{m,i}$와 $y_{m,j}$는 각각 기본 기억 Y_m의 i번째와 j번째 원소를 가리킨다.

  - 뉴런 사이의 시냅스 가중치를 행렬 형태로 나타내면 다음과 같다.

$$\mathbf{W} = \sum_{m=1}^{M} \mathbf{Y}_m \mathbf{Y}_m^T - M\mathbf{I}$$

  - 홉필드 신경망은 가중치 행렬이 대칭이다
  - 주 대각선이 모두 0이면 기본 기억 집합을 저장할 수 있다.

- 1단계: 저장
  - 홉필드 신경망의 행렬 : 식(6.23)

$$\mathbf{W} = \begin{bmatrix} 0 & w_{12} & \cdots & w_{1i} & \cdots & w_{1n} \\ w_{21} & 0 & \cdots & w_{2i} & \cdots & w_{2n} \\ \vdots & \vdots & & \vdots & & \vdots \\ w_{i1} & w_{i2} & \cdots & 0 & \cdots & w_{in} \\ \vdots & \vdots & & \vdots & & \vdots \\ w_{n1} & w_{n2} & \cdots & w_{ni} & \cdots & 0 \end{bmatrix} \qquad (6.23)$$

여기서 $w_{ij} = w_{ji}$다. 일단 가중치가 계산되면 바뀌지 않는다.

- 2단계 : 검사
  - 신경망에 기본 기억 y_m을 입력했다면 다시 불러 낼 수 있어야 한다

$$x_{m,i} = y_{m,i} \qquad i = 1, 2, \ldots, n \qquad m = 1, 2, \ldots, M$$

$$y_{m,i} = sign\left(\sum_{j=1}^{n} w_{ij}\, x_{m,j} - \theta_i \right)$$

06_홉필드 신경망

- 2단계 : 검사
  - $y_{m,i}$는 실제 출력 벡터 Y_m의 i번째 원소고, $x_{m,j}$는 입력 벡터 X_m의 j번째 원소다. 이를 행렬로 나타내면 다음과 같다.

$$\mathbf{X}_m = \mathbf{Y}_m \qquad m = 1, 2, \ldots, M$$
$$\mathbf{Y}_m = sign\left(\mathbf{W}\mathbf{X}_m - \theta\right)$$

  - 모든 기본 기억을 완벽하게 다시 불러낼 수 있으면 다음 단계를 진행한다.

- 3단계 : 복구
  - 알려지지 않은 n차원 검사 벡터 X를 신경망에 입력하고 안정 상태를 복구한다.
  - 보통 조금 잘못 되거나 불완전한 기본 기억을 검사 벡터로 사용한다.

$$\mathbf{X} \neq \mathbf{Y}_m \qquad m = 1, 2, \ldots, M$$

1. 홉필드 신경망의 복구 알고리즘을 초기화 한다.

$$x_j(0) = x_j \qquad j = 1, 2, \ldots, n$$

2. 각 뉴런의 초기 상태를 계산한다.

$$y_i(0) = sign\left(\sum_{j=1}^{n} w_{ij}\, x_j(0) - \theta_i\right) \qquad i = 1, 2, \ldots, n$$

- 3단계 : 복구

3. 이를 행렬 형태로 나타내면 반복 횟수 p=0에서의 상태 벡터는 다음과 같이 나타낼 수 있다.

$$\mathbf{Y}(0) = sign\,[\mathbf{W}\mathbf{X}(0) - \boldsymbol{\theta}]$$

4. 상태 벡터 Y(p)의 원소를 갱신한다.

$$y_i(p+1) = sign\left(\sum_{j=1}^{n} w_{ij}\, x_j(p) - \theta_i\right)$$

갱신되는 뉴런은 비동시적이다. 즉 한 번에 하나씩 임의로 선택한다.

5. 상태 벡터가 바뀌지 않을 때까지, 즉 안정 상태에 도달할 때까지 반복한다. 안정성 조건은 다음과 같이 정의할 수 있다.

$$y_i(p+1) = sign\left(\sum_{j=1}^{n} w_{ij}\, y_j(p) - \theta_i\right) \qquad i = 1, 2, \ldots, n \qquad (6.24)$$

6. 행렬로 표현한다.

$$\mathbf{Y}(p+1) = sign\,[\mathbf{W}\mathbf{Y}(p) - \boldsymbol{\theta}] \qquad (6.25)$$

06_홉필드 신경망

❖ 홉필드 신경망의 특징

- 홉필드 신경망 vs 인간의 기억

  [홉필드 신경망]

  - 홉필드 신경망은 기억의 자기 연상 유형을 표현한다.
  - 홉필드 신경망은 잘 못 되었거나 불완전한 기억을 복구할 수 있지만, 이로부터 다른 기억을 연상하지 못한다.

  [인간의 기억]

  - 인간의 기억은 연상이 가능하다.
  - 연상 사슬을 구축하고 이를 통해 잃어 버린 기억을 복구할 수 있다.

- 홉필드 신경망의 한계 및 극복 방법

  [한계]

  - 홉필드 신경망은 단층 신경망이다. 즉 입력 패턴을 적용한 뉴런 집단에서 바로 출력 패턴이 나온다.
  - 한 가지 기억에서 다른 기억을 연상하지 못 한다.

  [극복 방법]

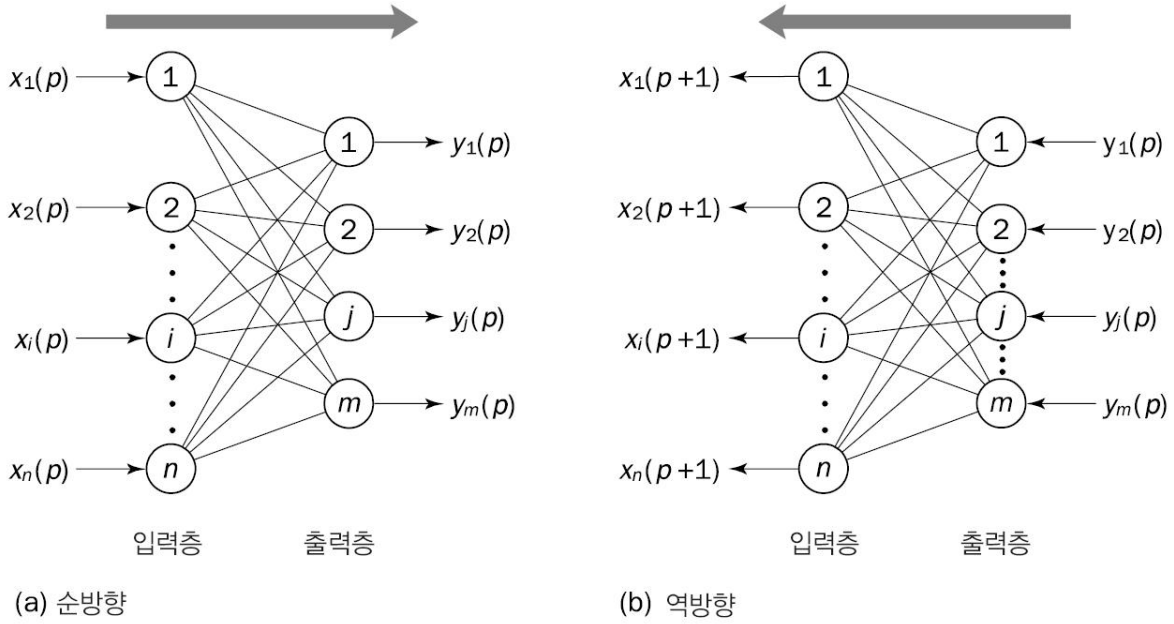
  - 한 가지 기억에서 다른 기억을 연상하려면 같지 않은 출력 패턴을 만들 수 있는 순환 신경망(recurrent neural network)을 이용한다.

07_양방향 연상 메모리

- 바트 코스코가 제안한 이형 연상 신경망이다.
- 한 집합(집합 A)의 패턴에서 다른 집합(집합 B)의 패턴을 연상할 수 있고, 그 반대도 가능하다.
- 홉필드 신경망처럼 일반화할 수 있고, 입력이 잘못되거나 불완전하더라도 올바른 출력을 내기도 한다.

❖ 양방향 연상 메모리의 동작

- **양방향 연상 메모리의 구조** : [그림 6-20]



[그림 6-20] BAM의 동작

07_양방향 연상 메모리

- **양방향 연상 메모리의 구조** : [그림 6-20]
  - [그림 6-20]은 기본적인 양방향 연상 메모리의 구조다.
  - 완전하게 연결된 층 2개(입력층, 출력층)로 이루어진다.
  - (a)와 같이 입력 벡터 X(p)를 가중치 행렬의 전치행렬 W^T에 적용해 출력 벡터 Y(p)를 만든다.
  - (b)와 같이 출력 벡터 Y(p)를 가중치 행렬 W에 적용해 새로운 입력 벡터 X(p + 1)를 만든다.
  - 이 과정을 입력과 출력 벡터가 변하지 않을 때까지, 즉 BAM이 안정 상태에 도달할 때까지 반복한다.
  - 양방향 연상 메모리 구조에서 가중치 행렬은 모든 상관관계 행렬의 합이다. : 식(6.29)

$$\mathbf{W} = \sum_{m=1}^{M} \mathbf{X}_m \mathbf{Y}_m^T \qquad (6.29)$$

M은 양방향 연상 메모리에 저장되는 패턴 쌍의 개수다.

❖ 양방향 연상 메모리의 훈련 알고리즘

- 홉필드 신경망처럼 보통 부호 활성화 함수를 따르는 맥클록-피츠 뉴런을 사용한다.
- 훈련 알고리즘은 총 3단계(저장, 검사, 복구)로 나눌 수 있다.

❖ 양방향 연상 메모리의 훈련 알고리즘

- 1단계: 저장
  - 양방향 연상 메모리는 패턴 쌍을 M개 저장한다.

예)

$$\text{집합 } A\text{: } \mathbf{X}_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad \mathbf{X}_2 = \begin{bmatrix} -1 \\ -1 \\ -1 \\ -1 \\ -1 \\ -1 \end{bmatrix} \quad \mathbf{X}_3 = \begin{bmatrix} 1 \\ 1 \\ -1 \\ -1 \\ 1 \\ 1 \end{bmatrix} \quad \mathbf{X}_4 = \begin{bmatrix} -1 \\ -1 \\ 1 \\ 1 \\ -1 \\ -1 \end{bmatrix}$$

$$\text{집합 } B\text{: } \mathbf{Y}_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \quad \mathbf{Y}_2 = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} \quad \mathbf{Y}_3 = \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix} \quad \mathbf{Y}_4 = \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix}$$

  - 이때 양방향 연상 메모리의 입력층과 출력층에는 뉴런이 각각 6개, 3개 있어야 한다.
  - 가중치 행렬 계산 시

$$\mathbf{W} = \sum_{m=1}^{4} \mathbf{X}_m \mathbf{Y}_m^T$$

07_양방향 연상 메모리

- 1단계: 저장
  - 가중치 행렬의 계산 예

$$\mathbf{W} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} + \begin{bmatrix} -1 \\ -1 \\ -1 \\ -1 \\ -1 \\ -1 \end{bmatrix} \begin{bmatrix} -1 & -1 & -1 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ -1 \\ -1 \\ 1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 & -1 & 1 \end{bmatrix}$$

$$+ \begin{bmatrix} -1 \\ -1 \\ 1 \\ 1 \\ -1 \\ -1 \end{bmatrix} \begin{bmatrix} -1 & 1 & -1 \end{bmatrix} = \begin{bmatrix} 4 & 0 & 4 \\ 4 & 0 & 4 \\ 0 & 4 & 0 \\ 0 & 4 & 0 \\ 4 & 0 & 4 \\ 4 & 0 & 4 \end{bmatrix}$$

- 2단계: 검사
  - 집합 A에서 벡터를 받았으면 집합 B에서, 집합 B에서 벡터를 받았으면 집합 A에서 관련 벡터를 불러올 수 있어야 한다. : 식(6.30)

$$\mathbf{Y}_m = sign\left(\mathbf{W}^T \mathbf{X}_m\right) \qquad m = 1, 2, \ldots, M \qquad (6.30)$$

- 2단계: 검사
  - 식(6.30) 계산 예

$$\mathbf{Y}_1 = sign\left(\mathbf{W}^T \mathbf{X}_1\right) = sign\left\{ \begin{bmatrix} 4 & 4 & 0 & 0 & 4 & 4 \\ 0 & 0 & 4 & 4 & 0 & 0 \\ 4 & 4 & 0 & 0 & 4 & 4 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \right\} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

  - Y_m이 주어졌을 때 양방향 연상 메모리가 X_m을 불러낼 있는지 확인한다. : 식(6.31)

$$\mathbf{X}_m = sign\left(\mathbf{W}\,\mathbf{Y}_m\right) \qquad m = 1, 2, \ldots, M \qquad (6.31)$$

식(6.31) 계산 예

$$\mathbf{X}_3 = sign\left(\mathbf{W}\,\mathbf{Y}_3\right) = sign\left\{ \begin{bmatrix} 4 & 0 & 4 \\ 4 & 0 & 4 \\ 0 & 4 & 0 \\ 0 & 4 & 0 \\ 4 & 0 & 4 \\ 4 & 0 & 4 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix} \right\} = \begin{bmatrix} 1 \\ 1 \\ -1 \\ -1 \\ 1 \\ 1 \end{bmatrix}$$

07_양방향 연상 메모리

- 3단계: 복구
  - 양방향 연상 메모리에 알려지지 않은 벡터(검사 벡터) X를 주고 저장된 연상을 구한다.
  - 검사 벡터로 보통BAM에 저장된 집합A(또는 집합 B)에 있는 패턴 중 하나를 골라 일부분을 변형시키거나 불완전하게 만들어서 사용한다.
  - Exclusive-OR 연산을 수행

$$\mathbf{X} \neq \mathbf{X}_m \qquad m = 1, 2, \ldots, M$$

1. 복구 알고리즘을 초기화 한다.

$$\mathbf{X}(0) = \mathbf{X} \qquad p = 0$$

2. p번째 반복할 때 양방향 연상 메모리의 출력을 계산한다.

$$\mathbf{Y}(p) = sign\left[\mathbf{W}^T \mathbf{X}(p)\right]$$

3. 입력 벡터 X(p)를 갱신한다.

$$\mathbf{X}(p+1) = sign\left[\mathbf{W}\,\mathbf{Y}(p)\right]$$

입력·출력 벡터가 바뀌지 않는 평형 상태에 이를 때까지 반복한다. 그러면 입력·출력 패턴은 서로 연관된 쌍을 표현할 것이다.

08_자기조직 신경망

- 자지조직 신경망은 교사나 감독 없이 학습할 수 있다.
- 역전파 신경망보다 훨씬 빠르게 학습하므로 실시간으로 쓰일 수 있다.

❖ 학습유형

- 감독 학습(supervised learning)
  - 관리되고 적극적인 학습
  - 교사 또는 훈련 집합을 신경망에 전달해 주는 학습

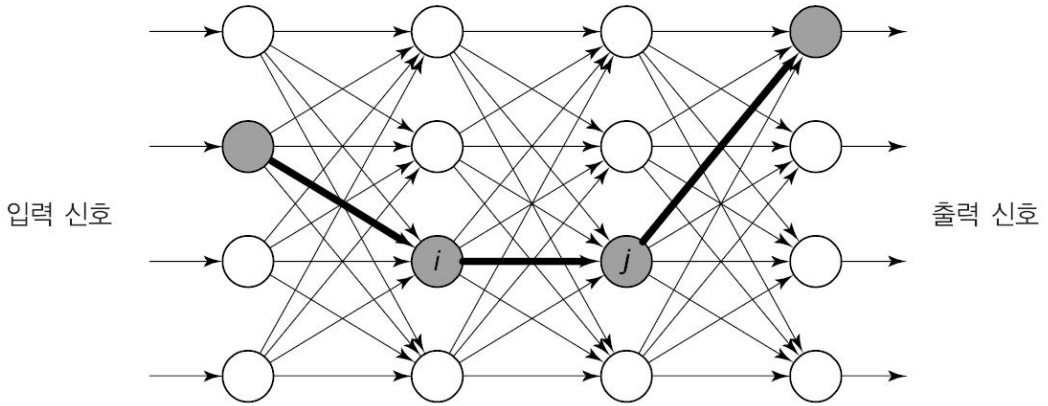
- 무감독 학습(self-organising neural network)
  - 모감독 또는 자기조직 학습이라고 부른다.
  - 외부 교사가 필요 없는 학습
  - 신경망은 훈련 기간 동안 각기 다른 여러 입력 패턴을 받고, 이 패턴에서 특징을 발견하여 입력 데이터를 적절한 범주로 분류하는 방법을 학습한다.
  - 무감독 학습은 뇌의 신경생물학적인 조직화를 따른다.
  - 무감독 학습 알고리즘은 빠르게 학습하는 것을 목표로 한다.
  - 무감독 학습의 예) 자기조직 신경망

❖ 헤브 학습: 자기조직 신경망

- 신경심리학자인 도널드 헤브가 제안

- **헤브의 법칙**
  - 뉴런 i가 뉴런 j를 흥분시킬 수 있고, 이런 과정을 반복적으로 수행하면 두 뉴런 간 시냅스 연결이 강화되고, 뉴런 j는 뉴런 i에서 오는 자극에 좀 더 민감해진다고 한다.
  - 층이 세 개인 역전파 신경망 : [그림 6-10]
  - 헤브 학습의 규칙

    1. 양쪽에 연결된 두 뉴런이 동시에 활성화되면 연결 가중치는 증가한다.
    2. 양쪽에 연결된 두 뉴런이 각자 활성화되면 연결 가중치는 감소한다.
  - 신경망에서 헤브 학습의 진행 예 : [그림 6-21]



[그림 6-21] 신경망에서의 헤브 학습

08_자기조직 신경망

- 헤브의 법칙
  - 헤브의 법칙을 이용해서 p번째 반복에서 가중치 w_{ij}에 적용되는 조정값을 식(6.32) 같이 표현할 수 있다.

$$\Delta w_{ij}(p) = F[y_j(p), x_i(p)] \qquad (6.32)$$

F[y_j(p), x_i(p)]는 시냅스 뒤와 앞의 활동에 대한 함수다.
헤브의 법칙을 다음과 같이 표현할 수 있다

$$\Delta w_{ij}(p) = \alpha y_j(p) x_i(p) \qquad (6.33)$$

α는 학습률 매개변수다. 이 식은 활동 생성 규칙이라 하며, 뉴런 쌍 사이에 시냅스 연결 가중치 변화가 들어오고 나가는 신호의 곱과 어떻게 연관되는지 보여준다.

- 망각 요소
  - 헤브 학습의 문제점: 가중치는 감소하지 않고 증가만 한다.
  - 헤브 학습의 문제점을 해결하려면 시냅스 가중치에 제한을 두어야 한다.
  - 헤브 학습에 비선형적인 망각 요소를 도입하여 이를 해결한다. : 식(6.34)

$$\Delta w_{ij}(p) = \phi y_j(p)[\lambda x_i(p) - w_{ij}(p)] \qquad (6.35)$$

φ가 망각 요소다.

08_자기조직 신경망

- 망각 요소
  - 망각 요소 φ는 하나의 학습 주기에서 가중치 감소를 정한다. 보통 0~1 사이 값이 들어간다.
  - 망각 요소가 0이면 가중치가 무한대로 증가한다. 반면에 망각 요소가 1에 가까우면 신경망은 학습한내용을 거의 기억하지 못한다.

❖ 헤브 학습 알고리즘

- 총 4단계로 구성되어 있음.
- 1단계: 초기화
  - 초기 시냅스 가중치와 임계값을 [0,1] 구간의 임의의 작은 값으로 설정한다. 또 학습률 매개변수 α와 망각요소 φ에 작은양(+)의 값을 할당한다.

- 2단계: 활성화
  - p번째 반복에서 뉴런 출력을 다음과 같이 계산한다.
  - 여기서 n은 뉴런의 입력 개수고, θ_j는 뉴런 j의 임계값이다.

$$y_j(p) = \sum_{i=1}^{n} x_i(p)\, w_{ij}(p) - \theta_j$$

❖ 헤브 학습 알고리즘

- 3단계: 학습
  - 신경망에 있는 가중치를 갱신한다.

$$w_{ij}(p+1) = w_{ij}(p) + \Delta w_{ij}(p)$$

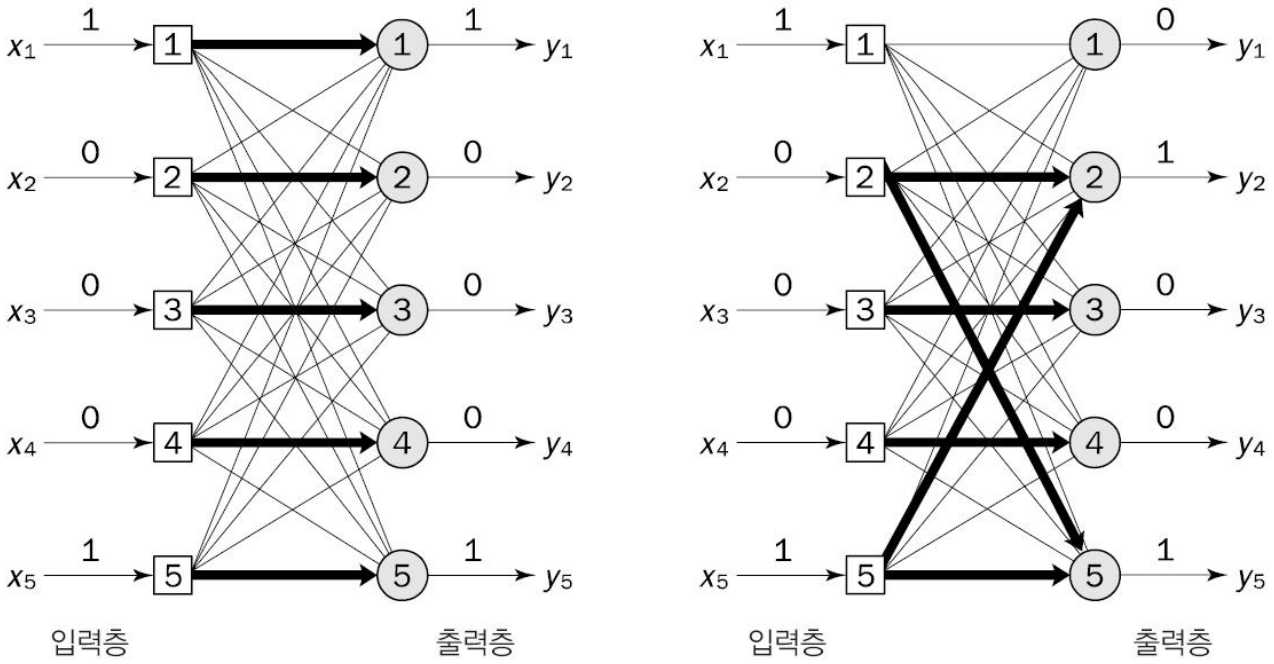
여기서 $\Delta w_{ij}(p)$는 p번째 반복 시 가중치 보정값이다. 다음과 같은 일반화된 활동 생성 규칙이 가중치 보정값을 결정한다.

$$\Delta w_{ij}(p) = \phi y_j(p)[\lambda x_i(p) - w_{ij}(p)]$$

- 4단계: 반복
  - 반복 횟수 p값을 1 증가시키고 2단계로 돌아가서 시냅스 가중치가 안정 상태에 이를 때까지 반복한다.

❖ 헤브 학습 알고리즘

- 단층 신경망에 대한 무감독 헤브 학습 예 : [그림 6-22]
  - 계산 뉴런이 5개인 단일 층을 가진 완전히 연결된 피드포워드 신경망



(a) 신경망의 초기와 최종 상태

❖ 헤브 학습 알고리즘

- 단층 신경망에 대한 무감독 헤브 학습 예: [그림 6-22]
  - 계산 뉴런이 5개인 단일 층을 가진 완전히 연결된 피드포워드 신경망

출력층

입력층	①	②	③	④	⑤
①	1	0	0	0	0
②	0	1	0	0	0
③	0	0	1	0	0
④	0	0	0	1	0
⑤	0	0	0	0	1

출력층

입력층	①	②	③	④	⑤
①	0	0	0	0	0
②	0	2.0204	0	0	2.0204
③	0	0	1.0200	0	0
④	0	0	0	0.9996	0
⑤	0	2.0204	0	0	2.0204

(b) 가중치 행렬의 초기와 최종 상태

[그림 6-22] 단층 신경망에 대한 무감독 헤브 학습

08_자기조직 신경망

❖ 경쟁 학습

- 경쟁학습의 기본 아이디어는 1970년대 초반에 소개되었다
- 처음에는 그다지 관심을 받지 못하다가 1980년대 후반 튜보 코호넨이 자기조직 특성 맵이라는 특수 인공 신경망을 소개하면서 관심이 높아졌다.

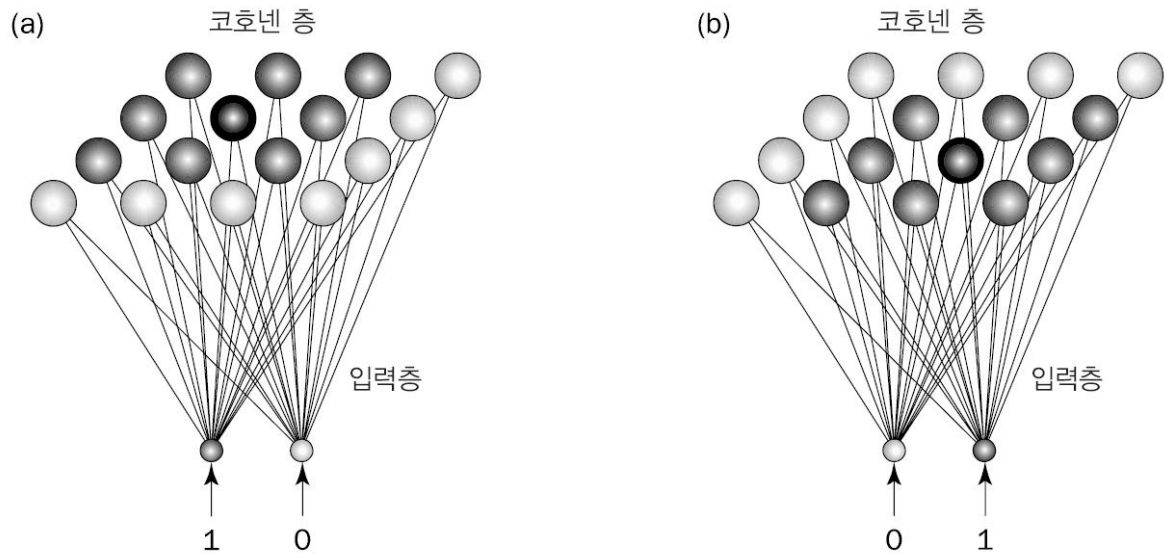
- 경쟁 학습(competitive learning)
  - 뉴런을 활성화하기 위해 뉴런들끼리 경쟁한다.
  - 헤브 학습에서는 출력 뉴런 몇 개를 동시에 활성화할 수 있으나 경쟁 학습에서는 항상 출력 뉴런 1개만 활성화할 수 있다.
  - '경쟁'에서 우승한 출력 뉴런을 승자 독식(winner-takes-all) 뉴런이라 한다.

- 자기조직 특성 맵
  - 인간의 뇌를 모델링한 것이다.
  - 뇌의 각 영역은 각기 다른 인간의 활동(운동, 시각, 청각, 이 외 감각 등)을 관장하고, 각기 다른 감각의 입력과 연결되어 있다. 각 감각의 입력이 대뇌피질의 해당 영역과 매핑되어 있다고 할 수 있다. 즉 대뇌피질은 인간의 뇌에 있는 자기조직 계산 맵이다.

- 자지조직 맵의 모델링
  - 코호넨의 지형도 형성 원칙과 특성 매핑 모델을 이용하여 모델링 한다. 이 모델은 뇌에 있는 자기조직 맵의 주요 성질에 착안하여 만들었으며, 컴퓨터로 쉽게 표현할 수 있다.

03_요약

- 자지조직 맵의 모델링
  - 코호넨 모델은 입력층에서 일정 개수의 입력 패턴을 받아 조금 더 고차원의 출력층, 즉 코호넨 층에 놓음으로써 위상적인 매핑을 제공한다. : [그림 6-23]



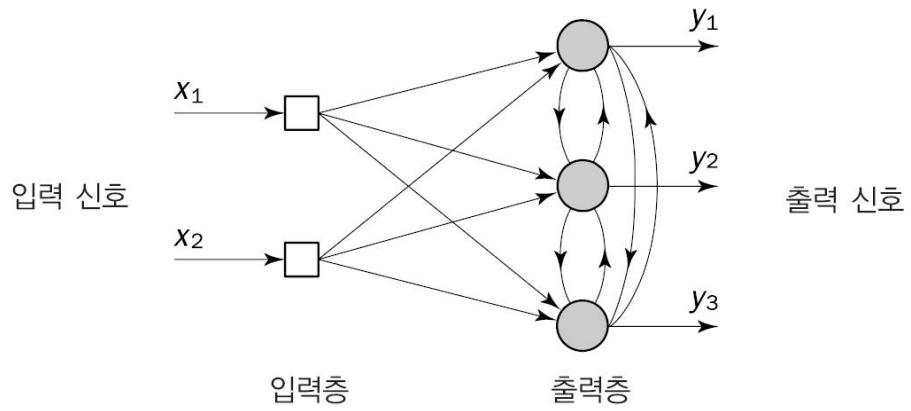
[그림 6-23] 특성 매핑 코호넨 모델

코호넨 층은4×4 뉴런으로 만들어진2차원 격자로 이루어져 있는데, 뉴런 각각에는 입력이 2개 들어간다.

우승한 뉴런은 검은 색, 그 이웃은 회색으로 표시한다. 여기서 승자의 이웃이란 물리적으로 승자쪽에 가까운 뉴런을 말한다.

08_자기조직 신경망

- 코호넨 신경망
  - 계산 뉴런이 단일 층으로 구성되어 있지만, 유형이 다른 연결이 두 개 있다. : [그림 6-24]



[그림 6-24] 코호넨 신경망의 구조

  - 입력층 뉴런에서 출력층 뉴런으로 가는 순방향 연결과 출력층 뉴런 간 측방향 연결이 있다. 측방향 연결은 뉴런 간 경쟁을 생성하는 데 쓰인다.
  - 모든 뉴런 중 활성화 수준이 가장 높은 뉴런이 승자가 된다(승자 독식 뉴런). 이 뉴런은 출력 신호를 생성하는 유일한 뉴런이다.
  - 경쟁에서 진 다른 뉴런들의 활성은 억제된다.

❖ 경쟁학습 알고리즘

- 총 4단계로 구성되어 있음
- 1단계: 초기화
  - 초기 시냅스 가중치를 임의의 값, 예를 들어 [0,1] 구간 내 값으로 설정하고, 학습률 매개변수 α에 작은 양의 값을 할당한다.
- 2단계: 활성화와 유사도 매핑
  - 입력 벡터 X를 적용하여 코호넨 신경망을 활성화하고, 다음 식과 같은 최소 거리 유클리드 기준을 적용하여 p번째 반복에서 승자 독식 뉴런 jX를 찾는다.
  - 여기서 n은 입력층에 있는 뉴런 개수고,m은 출력층, 즉 코호넨 층의 뉴런 개수다.

$$j_{\mathbf{X}}(p) = \min_{j} \|\mathbf{X} - \mathbf{W}_j(p)\| = \left\{ \sum_{i=1}^{n} [x_i - w_{ij}(p)]^2 \right\}^{1/2} \qquad j = 1, 2, \ldots, m$$

- 3단계 : 학습
  - 시냅스 가중치를 다음과 같이 갱신한다. $\Delta w_{ij}(p)$는 p번째 반복에서의 가중치 보정값이다.

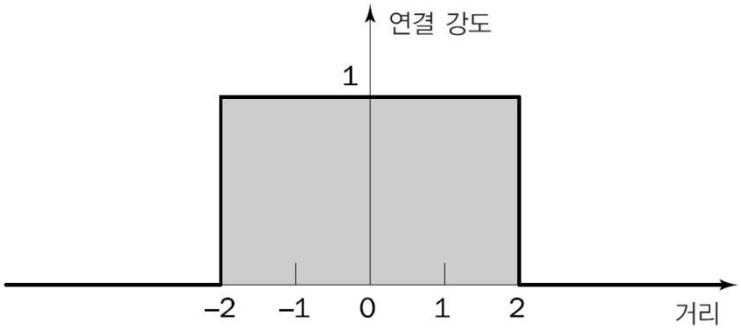
$$w_{ij}(p+1) = w_{ij}(p) + \Delta w_{ij}(p)$$

- 3단계 : 학습
  - 경쟁 학습 규칙으로 가중치 보증값을 결정한다.

$$\Delta w_{ij}(p) = \begin{cases} \alpha[x_i - w_{ij}(p)] & j \in \Lambda_j(p) \\ 0 & j \notin \Lambda_j(p) \end{cases} \qquad (6.39)$$

여기서 α는 학습률 매개변수고, $\wedge_j(p)$는 p번째 반복에서 승자 독식 뉴런 j_X를 중심으로 한 이웃함수를 나타낸다.

  - $\wedge_j$는 보통 상수 범위다. 위상적으로 이웃 범위에 속한 모든 뉴런이 동시에 깨어나고, 뉴런 간 관계는 승자 독식 뉴런 jX와의 거리와 무관하다.
  - 이웃 함수의 형태 : [그림 6-27]



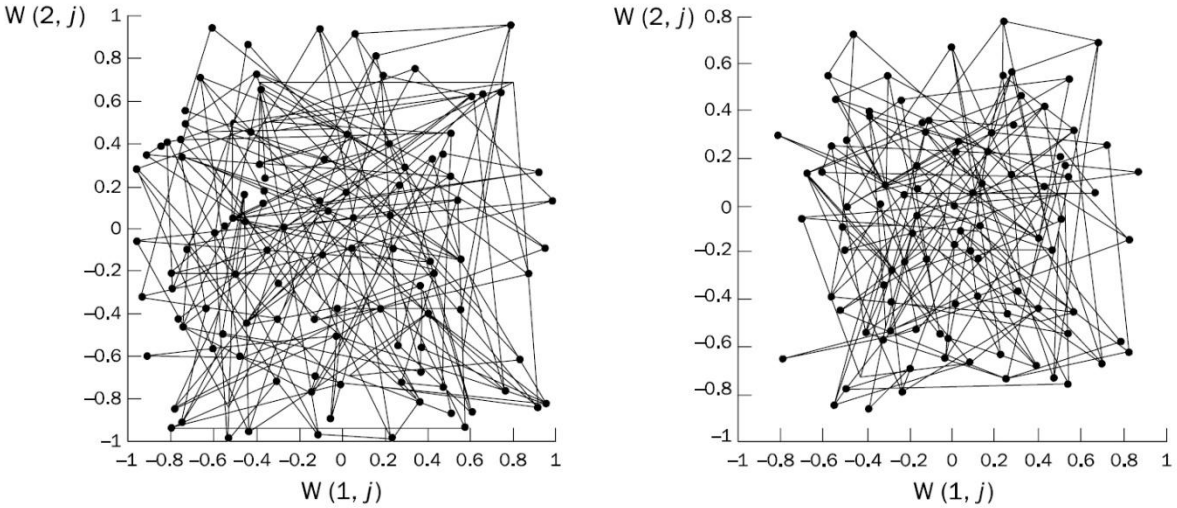
[그림 6-27] 직사각형 이웃 함수

$$y_j = \begin{cases} 1 & j \in \Lambda_j(p) \\ 0 & j \notin \Lambda_j(p) \end{cases}$$

08_자기조직 신경망

- 4단계 : 반복
  - 반복 횟수 p 값을 1 증가시키고 2단계로 돌아간다. 최소 거리 유클리드 기준을 만족하거나 특성 맵에서 의미 있는 변화가 일어나기 전까지 반복한다..

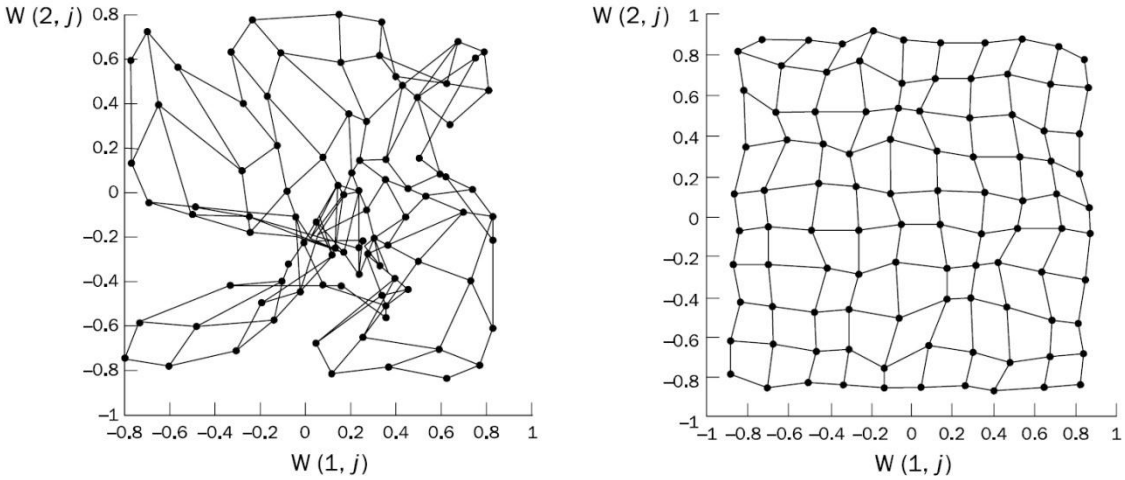
- 코호넨 신경망의 경쟁 학습 예: [그림 6-28]
  - 2차원 격자 형태(10×10)로 뉴런 100개가 나열된 코호넨 신경망.
  - 신경망은 2차원 입력 벡터를 분류한다. 즉 신경망에 있는 각 뉴런은 자신의 영역에서 일어난 입력 벡터에만 반응한다.
  - 신경망은 -1~+1구간의 정사각형 영역에서 임의로 생성된 2차원 입력 벡터 1,000개로 훈련 받는다. 초기 시냅스 가중치에도-1~+1구간에 있는 임의의 값이 들어가며, 매개변수 α는 0.1이다.



(a) 임의의 초기 가중치 (b) 100회 반복한 후의 신경망

08_자기조직 신경망

- **코호넨 신경망의 경쟁 학습 예: [그림 6-28]**

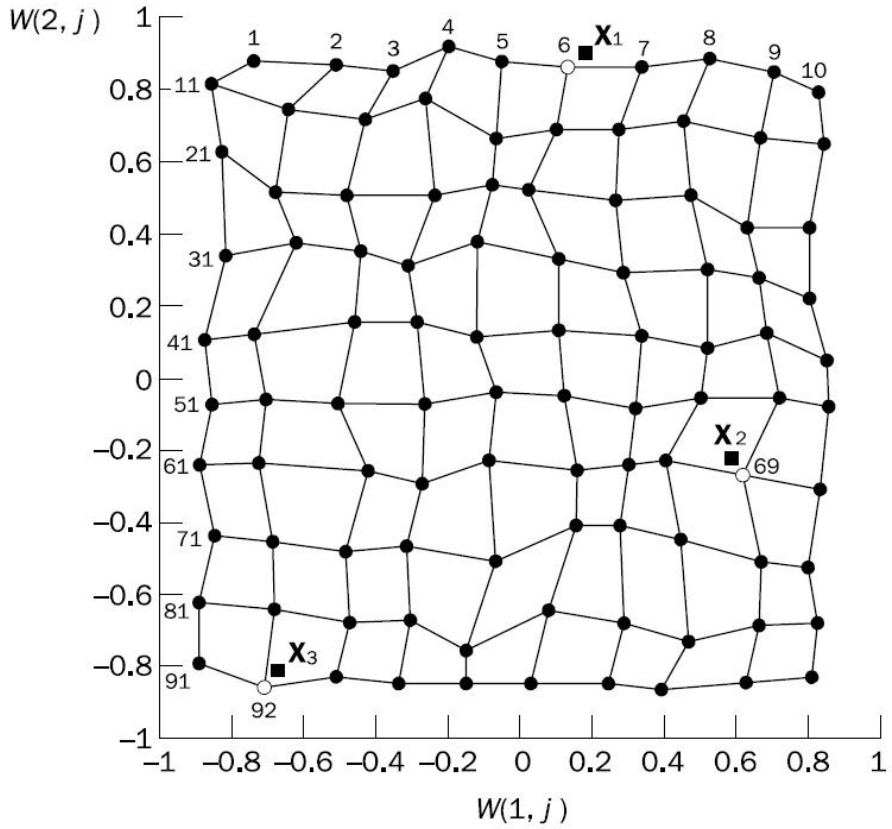


(c) 1,000회 반복한 후의 신경망 (d) 10,000회 반복한 후의 신경망

[그림 6-28] 코호넨 신경망의 경쟁 학습

- **뉴런 6은 입력벡터X1에, 뉴런 69는 입력 벡터X2에, 뉴런 92는 입력벡터 X_3에 반응한다. 따라서 입력 공간에 그려진 특성 맵은 위상적으로 정렬되었으며, 격자에서 뉴런의 공간적 위치는 입력 패턴의 특정 성질과 일치한다.**

- 코호넨 신경망의 경쟁 학습 예
  - 입력 공간에 그린 위상적으로 정렬된 특성 맵. : [그림 6-29]



[그림 6-29] 입력 공간에 그린 위상적으로 정렬된 특성 맵

09_요약

❖ 인공 신경망 요약

- 기계 학습
  - 기계 학습은 컴퓨터가 경험을 통해 학습하고, 예를 통해 배우며, 유추하여 학습하게 만드는 적응 메커니즘과 관련되어 있다. 학습 능력은 시간이 지나면서 지능형 시스템의 성능을 개선한다. 기계 학습에서 가장 대중적인 접근법 중 하나는 인공 신경망이다.
  - 워렌 맥클록과 월터 피츠는 대다수 인공 신경망의 기초가 되는 아주 간단한 아이디어를 제안했다. 뉴런은 입력 신호의 가중합을 계산하고. 그 결과를 임계값 θ와 비교한다. 순 입력이 임계값보다 작으면 뉴런의 출력 값은 -1이다. 그러나 순 입력이 임계값과 같거나 크면 뉴런을 활성화되고, 출력 값은 +1이다.

- 퍼셉트론
  - 프랭크 로젠블랫은 가장 간단한 신경망 형태인 퍼셉트론을 제안했다. 퍼셉트론의 동작 원리는 맥클록-피츠 뉴런 모델을 바탕으로 하며, 조정 가능한 시냅스 가중치와 하드 리미터로 구성된다. 또한 학습은 실제 출력과 목표 출력 간 격차를 줄이도록 가중치를 조절하는 방법으로 진행된다. 초기 가중치는 임의로 할당되었지만, 이후 훈련 예제와 일치하는 출력을 얻도록 갱신된다.
  - 퍼셉트론은 선형으로 분리할 수 있는 함수만 학습할 수 있으며, 지역적으로 학습된 예제로는 전체적인 일반화를 할 수 없다. 역전파 알고리즘으로 학습한 다층 퍼셉트론 같은 발전된 형태의 신경망을 사용하면 로젠블랫 퍼셉트론의 한계를 극복할 수 있다.

09_요약

- 다층 신경망
  - 다층 신경망은 공급 뉴런으로 이루어진 입력층, 계산 뉴런으로 이루어진 하나 이상의 중간 또는 은닉층, 계산 뉴런으로 이루어진 출력층이 있는 피드포워드 신경망이다. 입력층은 외부에서 입력 신호를 받고, 이 신호를 은닉층에 있는 모든 뉴런에 재분배한다. 은닉층에 있는 뉴런은 특성을 파악한다. 뉴런의 가중치는 입력 패턴에 숨겨져 있는 특성을 나타낸다. 출력층은 전체 신경망의 출력 패턴을 정한다.
  - 다층망의 학습 방법은 퍼셉트론의 학습 방법과 같다. 알고리즘에는 두 단계가 있다. 먼저 훈련입력 패턴을 신경망의 입력층에 전달한다. 그런 다음 신경망은 출력층에서 출력 패턴이 생성될 때까지 각 층에 입력 패턴을 전파한다. 만약 출력 패턴이 목표 패턴과 다르면 오차를 계산한 후 출력층에서 입력층으로 신경망을 따라 거꾸로 전파한다. 오차가 전파되면서 가중치가 수정된다.

- 역전파 학습
  - 역전파 학습은 널리 쓰이고 있지만, 계산 부담이 크고 학습이 느리기 때문에, 실제 응용할 때는 순수한 역전파 알고리즘을 거의 사용하지 않는다. 역전파 알고리즘의 계산 효율을 높이는 몇 가지 방법이 있다. 다층 신경망은 시그모이드 활성화 함수가 쌍곡 탄젠트로 표현될 때 좀 더 빠르게 학습한다. 운동량과 적응 학습률도 다층 역전파 신경망의 성능을 높여준다.

09_요약

- 홉필드 신경망
  - 홉필드 신경망의 훈련 알고리즘에는 저장과 복구라는 2가지 기본 단계가 있다. 첫 번째 단계에서 신경망은 상태의 집합 또는 모든 뉴런의 현재 출력으로 결정된 기본 기억을 저장해야 한다. 이는 신경망의 가중치 행렬을 계산함으로써 이루어진다. 일단 가중치를 계산하면 그대로 유지한다. 두 번째 단계에서는 기본 기억 중 잘못되거나 불완전한 입력이 신경망에 들어간다. 입력을조정하도록 신경망의 출력을 계산하고 피드백한다. 출력값이 상수가 될 때까지 이 과정을 반복한다. 기본 기억을 제대로 복구하려면 홉필드 신경망의 저장 용량을 작게 잡아야 한다.

- 양방향 연상 메모리
  - 양방향 연상 메모리의 기본 아이디어는 집합 A에 있는 n차원 벡터X가 입력으로 들어왔을 때는 집합 B에 있는 m차원 벡터 Y를 불러내고, Y가 입력으로 들어왔을 때는 양방향 연상 메모리이 X를 불러내도록 패턴 쌍을 저장하는 것이다. 홉필드 신경망의 저장 용량을 제한하는 조건은 양방향 연상 메모리에도 적용할 수 있다. 양방향 연상 메모리에 저장되는 최대 연상 개수는 더 작은 층의 뉴런 개수를 초과하면 안 된다. 또 다른 문제는 잘못된 수렴이다. 양방향 연상 메모리는 가장 근접한 연상을 생성하는 것은 아니다.

09_요약

- 무감독 학습
  - 훈련 집합을 신경망에 전달하는 '교사' 가 있는 학습과 달리, 무감독 또는 자기조직 학습에는 교사가 필요가 없다. 신경망은 훈련 기간 동안 각기 다른 여러 입력 패턴을 받고, 이 패턴에서 특징을 발견하여 입력 데이터를 적절한 범주로 분류하는 방법을 학습한다.
  - 헤브의 법칙(Hebb's law): 다. 헤브의 법칙에 따르면, 뉴런 i가 뉴런 j를 흥분시킬 수 있고, 이런 과정을 반복적으로 수행하면 두 뉴런 간 시냅스 연결이 강화되고, 뉴런 j는 뉴런 i에서 오는 자극에 좀 더 민감해진다고 한다. 이 법칙은 교사가 없는 학습의 기초가 된다. 여기서의 학습은 환경으로부터 피드백을 받지 않는 국지적인 현상으로 볼 수 있다.
  - 경쟁 학습: 경쟁 학습에서 뉴런들은 활성화되기위해 그들끼리 경쟁한다.'경쟁' 에서 우승한 출력 뉴런을 승자 독식(winner-takes-all) 뉴런이라 한다. 경쟁 학습의 기본 아이디어는 1970년대 초반에 소개되었지만, 처음에는 그다지 관심을 받지 못하다가 1980년대 후반 튜보 코호넨이 자기조직 특성 맵이라는 특수한 종류의 인공신경망을 소개하면서 관심이 높아졌다. 또한 코호넨은 지형도상의 출력 뉴런의 공간적 위치가 입력 뉴런의 특정 성질에 대응된다는 지형도 형성 원칙을 정식화했다.

- 코호넨 신경망
  - 코호넨 신경망은 계산 뉴런을 담은 단일 층으로 이루어져 있지만, 유형이 다른 연결이 두 개 있다. 입력층 뉴런에서 출력층 뉴런으로 가는 순방향 연결과 출력층 뉴런 간 측방향 연결이 있다. 측방향 연결은 뉴런 간 경쟁을 생성하는 데 쓰인다. 코호넨 신경망에서 뉴런은 가중치를 활성화되지 않은 연결에서 활성화된 연결로 옮겨서 학습한다. 학습은 승자 뉴런과 그 이웃만 할 수 있다. 주어진 입력에 반응하지 않는 뉴런은 학습할 수 없다.

09_요약

- 홉필드 신경망
  - 홉필드 신경망의 훈련 알고리즘에는 저장과 복구라는 2가지 기본 단계가 있다. 첫 번째 단계에서 신경망은 상태의 집합 또는 모든 뉴런의 현재 출력으로 결정된 기본 기억을 저장해야 한다. 이는 신경망의 가중치 행렬을 계산함으로써 이루어진다. 일단 가중치를 계산하면 그대로 유지한다. 두 번째 단계에서는 기본 기억 중 잘못되거나 불완전한 입력이 신경망에 들어간다. 입력을조정하도록 신경망의 출력을 계산하고 피드백한다. 출력값이 상수가 될 때까지 이 과정을 반복한다. 기본 기억을 제대로 복구하려면 홉필드 신경망의 저장 용량을 작게 잡아야 한다.

- 양방향 연상 메모리
  - 양방향 연상 메모리의 기본 아이디어는 집합 A에 있는 n차원 벡터X가 입력으로 들어왔을 때는 집합 B에 있는 m차원 벡터 Y를 불러내고, Y가 입력으로 들어왔을 때는 양방향 연상 메모리이 X를 불러내도록 패턴 쌍을 저장하는 것이다. 홉필드 신경망의 저장 용량을 제한하는 조건은 양방향 연상 메모리에도 적용할 수 있다. 양방향 연상 메모리에 저장되는 최대 연상 개수는 더 작은 층의 뉴런 개수를 초과하면 안 된다. 또 다른 문제는 잘못된 수렴이다. 양방향 연상 메모리는 가장 근접한 연상을 생성하는 것은 아니다.

Thank You !