

개요

● 신경회로망(Neural Networks)

□ 생물학적 시스템의 계산틀 → 컴퓨터로 모방 불가 → 생물학적 정보체계 연구를 위해 시작 - 심리학,신경과학,인지과학, 시스템 이론 병합

□ 기본 작업

- 학습(**learning**): 패턴 부류에 따라 신경망의 연결가중치 조정
- 재생(**recall**): 학습된 가중치와 입력벡터와의 거리 계산하여 가장 가까운 클래스로 분류

→ 사람과 같은 학습 능력: 패턴 분류, 인식, 최적화, 예측

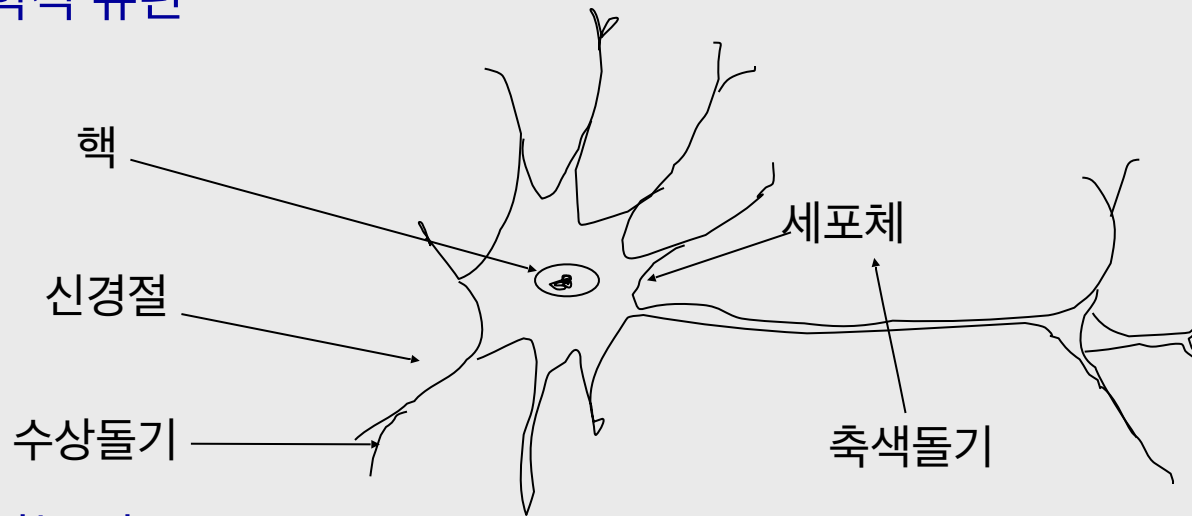
→ 기존 인공지능 문제 해결의 새로운 계산틀 제공

□ (인공)신경회로망

- 인간의 두뇌작용을 신경 세포들간의 연결관계로 모델링
→ 인간의 학습을 모델링

개요

□ 생물학적 뉴런



□ 신경기능 탐구

- 생물학적 신경회로망: 인간의 생체 기능 연구(생물, 생리, 신경과학자) → 동물 생체 실험 → 학습능력의 원천 밝힘
- 인공신경회로망 - 신경회로망의 기능, 구조를 **H/W, S/W**적으로 실현 → 수학적 모델 연구(기존의 시뮬레이션으로 구현된 알고리즘을 **H/W**화)

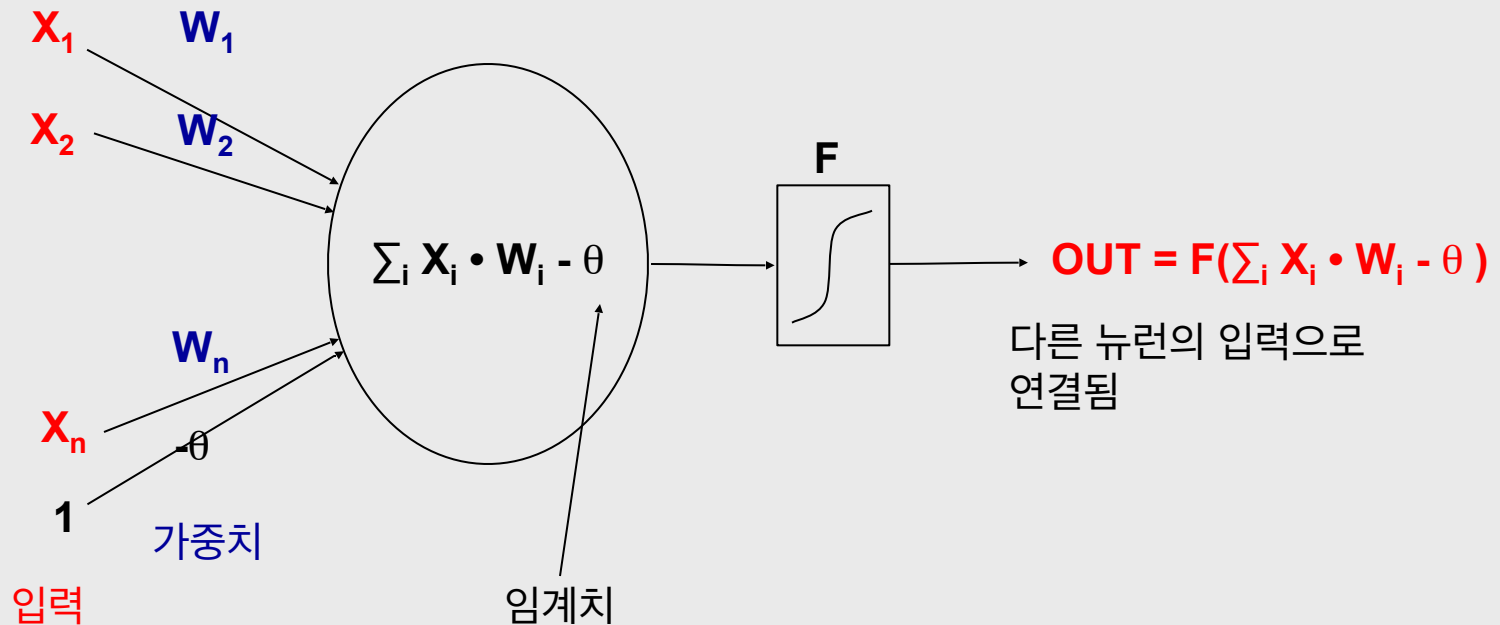
역사

- (1943) **McCulloch**와 **Pitts** - 신경세포들을 단순 연결한 신경회로망은 산술 및 논리연산을 수행할 수 있다.
 - (1949) **Hebb**의 법칙
 - 최초의 인공 신경회로망의 학습규칙 제안
 - 신경회로망이 학습을 수행함을 증명
 - 50, 60년대
 - 단층의 인공세포의 신경회로망
 - **Rosenblatt, Widrow**의 **Perceptron**- 일기예보, 심전도 해석, 인공시각
 - 69년 **Minsky** 등에 의해 한계 지적(퍼셉트론은 **XOR** 문제해결 불가)
 - 신경망 연구 침체 → 기호주의 **AI** 활성화
 - 80년대 중반이후
 - 기호주의 **AI** 한계
 - **Hopfield**: 신경회로망 ← 에너지 개념
 - 오류역전파 학습 알고리즘(**Rumelhart, McClelland**): 다층 퍼셉트론 학습
- 신경회로망 연구 부흥

정의 및 특징

● 신경회로망의 정의, 특징

- 신경 세포가 신경절로 연결되어 정보전달
- 노드 또는 처리요소(**processing element**)를 연결 → 신경회로망
- 인공 신경회로망의 뉴런(처리요소)의 구조
 - 뉴런1개는 단순기능 → 다중으로 연결되면 강력한 기능



정의 및 특징

- 신경회로망이 커지고 복잡해질수록 더 나은 기능 수행
 - 다층 신경회로망: 입력층과 출력층 사이에 새로운 층 추가
→ 은닉층(**hidden layer**) 또는 중간층(**internal layer**)
- 예를 통한 학습, 일반화, 연상기억, 결함 허용성의 특징을 보인다
 - 예를 통한 학습: 예를 계속적으로 제시함으로써 원하는 형태의 사상(**mapping**)을 형성 (지도 학습/비지도 학습)
 - 일반화: 학습이 완료된 신경회로망은 학습되지 않은 입력에 대해서도 올바른 결과를 출력
 - 연상기억: 새로운 입력, 일부 유실된 정보 → 유사한 출력
 - 결함 허용성: 일부 뉴런 고장, 단절 → 남아있는 뉴런에 의해 작동보장
- 연결가중치 조정방법
 - 지도학습: 입력이 주어짐에 따라 원하는 출력값이 활성화 되도록 가중치를 조절 (**Hopfield** 학습규칙, 델타규칙, 오류 역전파 학습규칙)
 - 비지도학습: 목표값 없이 학습 데이터만 입력, 스스로 연결가중치들을 학습 → 미리 결정된 해가 불필요 (경쟁학습, **ART** 모델)

종류

● 신경회로망 모델 종류

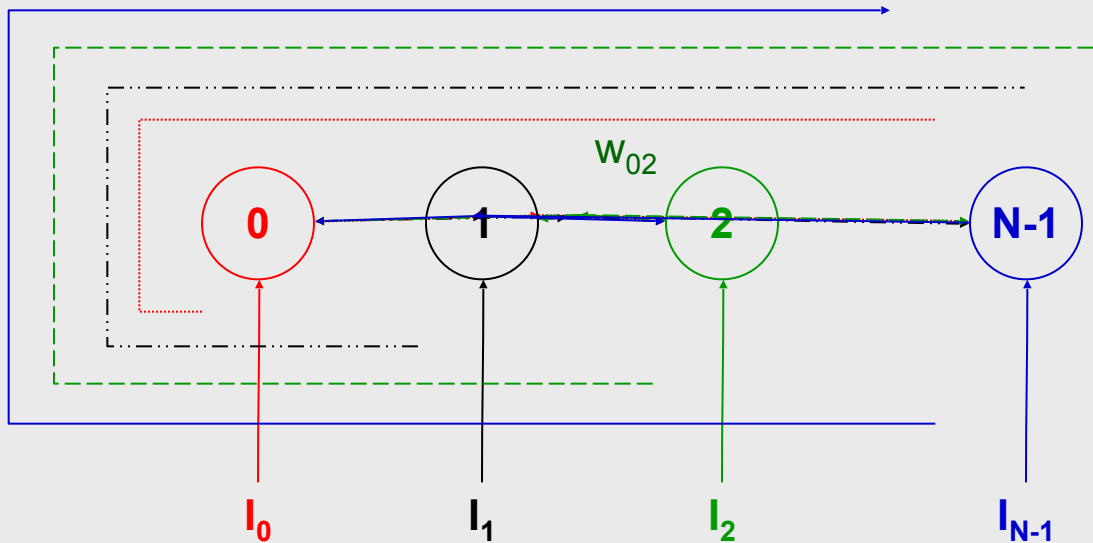
| 입력형식 | 학습방식 | 신경회로망 모델 |
|------|--------------------|--|
| 이진입력 | 지도 학습 | • Hopfield network |
| | 지도 학습 및 비지도 학습의 결합 | • Counter-Propagation Network |
| | 비지도 학습 | • ART Model |
| 실수입력 | 지도 학습 | • Perceptron • Multilayer Perceptron |
| | 비지도 학습 | • Competitive learning • Self-Organization Map(SOM) |

Hopfield 신경회로망

□ 신경회로망의 동작: **energy + dynamic system**

□ 조직

- 모든 뉴런이 양방향으로 연결(자신 제외)
- 전달(활성화)함수 : **hard limiter** 함수 사용
- 입력 : 2진수
- 출력 : **+1, -1**(양과 음의 에너지 상태를 표시)



Hopfield 신경회로망의 동작 규칙

□ 1단계 : **M**개의 패턴을 이용하여 **N**개 뉴런 사이의 연결 가중치 지정

$$w_{ij} = \begin{cases} \sum_{s=0}^{M-1} X_i^s X_j^s, & i \neq j \\ 0, & i = j \end{cases} \quad (0 \leq i, j \leq N-1)$$

- w_{ij} : 뉴런 **i**에서 뉴런 **j**로의 연결가중치
- X_i^s : **s** 클래스에 속하는 학습패턴의 **i**번째 요소값(+1 or -1)

□ 2단계 : 미지의 입력패턴을 신경회로망에 제시

$$\mu_i(0) = x_i, \quad (0 \leq i, j \leq N-1)$$

□ 3단계 : 뉴런들의 출력과 가중치를 곱한 값을 합하여 전달함수 통과

$$\mu_j(t+1) = f_h\left(\sum_{i=0}^{N-1} \mu_i(t) \cdot w_{ij}\right), \quad (0 \leq j \leq N-1) \quad \text{where} \quad f_h(a) = \begin{cases} 1, & \text{if } a \geq 0 \\ -1, & \text{if } a < 0 \end{cases}$$

□ 4단계 : 수렴(출력의 변화가 없음)할 때까지 3단계를 계속 반복

□ 5단계 : 2단계로 분기

Hopfield 신경회로망 - 비교

□ 예제

2개의 4차원 학습패턴벡터: $P_0=(1,1,1,-1)$, $P_1=(1,-1,1,-1)$ 일때 w_{ij} 는?

- $w_{00} = w_{11} = w_{22} = w_{33} = 0$
- $w_{01} = X_0^0 X_1^0 + X_0^1 X_1^1 = 1 \times 1 + 1 \times (-1) = 0$

$$w_{ij} = \begin{bmatrix} 0 & 0 & 2 & -2 \\ 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & -2 \\ -2 & 0 & -2 & 0 \end{bmatrix}$$

→ 새로운 입력 **(1, 1, -1, -1)** 주어질 때 어떤 패턴으로 인식되는가?

$$\begin{aligned} \mu(0) : (1, 1, -1, -1) & \quad \mu_2(1) = f_h(\mu_0(0) \cdot w_{02} + \mu_1(0) \cdot w_{12} + \mu_2(0) \cdot w_{22} + \mu_3(0) \cdot w_{32}) \\ \mu(1) : (1, 1, 1, 1) & \quad = f_h(1 \times 2 + 1 \times 0 + (-1) \times 0 + (-1) \times (-2)) = 1 \\ \mu(2) : (1, 1, 1, -1) & \\ \mu(3) : (1, 1, 1, -1) & \quad \text{3회 재생후 수렴} \rightarrow \mathbf{P_0} \text{로 인식} \end{aligned}$$

□ 예제: 각 패턴 크기: (5x5) 대상 : {ㄱ, ㄴ, ㄷ, ㄹ}으로 학습

• 학습 패턴

| | | | | |
|----|----|----|----|---|
| 1 | 1 | 1 | 1 | 1 |
| -1 | -1 | -1 | -1 | 1 |
| -1 | -1 | -1 | -1 | 1 |
| -1 | -1 | -1 | -1 | 1 |
| -1 | -1 | -1 | -1 | 1 |

| | | | | |
|---|----|----|----|----|
| 1 | -1 | -1 | -1 | -1 |
| 1 | -1 | -1 | -1 | -1 |
| 1 | -1 | -1 | -1 | -1 |
| 1 | -1 | -1 | -1 | -1 |
| 1 | 1 | 1 | 1 | 1 |

| | | | | |
|---|----|----|----|----|
| 1 | 1 | 1 | 1 | 1 |
| 1 | -1 | -1 | -1 | -1 |
| 1 | -1 | -1 | -1 | -1 |
| 1 | -1 | -1 | -1 | -1 |
| 1 | 1 | 1 | 1 | 1 |

| | | | | |
|----|----|----|----|----|
| 1 | 1 | 1 | 1 | 1 |
| -1 | -1 | -1 | -1 | 1 |
| 1 | 1 | 1 | 1 | 1 |
| 1 | -1 | -1 | -1 | -1 |
| 1 | 1 | 1 | 1 | 1 |

• 테스트 패턴 → {ㄱ, ㄴ, ㄷ, ㄹ}으로 테스트

– ㄱ, ㄴ, ㄷ: 제대로 인식

– ㄹ:

| | | | | |
|----|----|----|----|----|
| 1 | 1 | 1 | 1 | 1 |
| -1 | -1 | -1 | -1 | 1 |
| 1 | 1 | 1 | 1 | 1 |
| 1 | -1 | -1 | -1 | -1 |
| 1 | 1 | 1 | 1 | 1 |



| | | | | |
|----|----|----|----|----|
| 1 | 1 | 1 | 1 | 1 |
| -1 | -1 | -1 | -1 | 1 |
| 1 | -1 | -1 | -1 | -1 |
| 1 | -1 | -1 | -1 | -1 |
| 1 | 1 | 1 | 1 | 1 |



| | | | | |
|---|----|----|----|----|
| 1 | 1 | 1 | 1 | 1 |
| 1 | -1 | -1 | -1 | -1 |
| 1 | -1 | -1 | -1 | -1 |
| 1 | -1 | -1 | -1 | -1 |
| 1 | 1 | 1 | 1 | 1 |



| | | | | |
|---|----|----|----|----|
| 1 | 1 | 1 | 1 | 1 |
| 1 | -1 | -1 | -1 | -1 |
| 1 | -1 | -1 | -1 | -1 |
| 1 | -1 | -1 | -1 | -1 |
| 1 | 1 | 1 | 1 | 1 |

ㄷ 으로 오인식

– 회로망 크기 vs 저장가능한 패턴수 문제

→ Hopfield에서는 뉴런 수가 N인 경우, $0.15N$ 개의 패턴 저장 가능
(이 예제의 경우 $25 \times 0.15 = 3.75 \rightarrow 4$ 개 미만의 패턴 인식)

• 수렴결과가 최적인지 보장 안됨

단층 퍼셉트론(Perceptron)

- 눈의 망막을 모델화함(Rosenblatt)
- 지도학습, 이진 & 아날로그 입력처리
- 알고리즘 : 전체 출력뉴런들에 대하여 계산된 출력값과 목표값과의 차이를 최소화시킴(Widrow-Hoff 규칙(델타규칙)이 유명)
→만일 계산된 출력값과 목표값간에 차이가 없으면 연결 가중치는 변경되지 않으며,차이가 있으면 차이를 줄이는 방향으로 가중치를 변경.

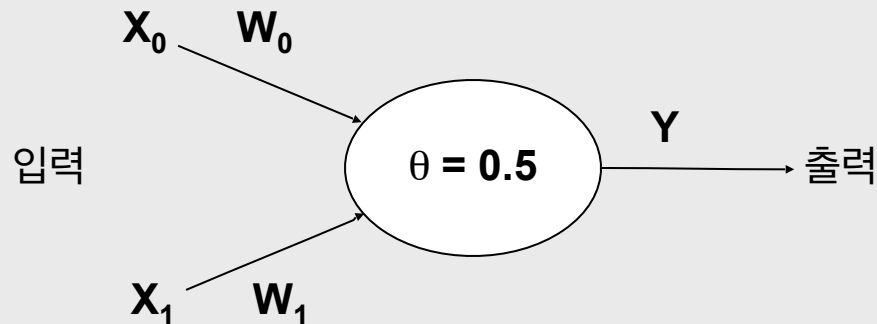
□ 학습단계

- 1단계. 가중치($W_i(0)$)와 임계치(θ)를 초기화.
- 2단계. 새로운 입력패턴(X_0, X_1, \dots)과 목표출력 패턴($d(t)$)을 제시
- 3단계. 하드리미터 함수를 사용하여 실제 출력값($y(t)$)을 계산

$$y(t) = f_h\left(\sum_{i=0}^{N-1} W_i(t) X_i(t) - \theta\right)$$

- 4단계. 가중치를 갱신 $W_i(t+1) = W_i + \alpha[d(t) - y(t)] \cdot X_i(t), (0 \leq i \leq N-1)$
- 5단계. 2단계로 분기하여 반복 수행

□ AND, XOR 예제



| 입력 | | 출력 | | |
|-------|-------|-----|---|-----|
| X_0 | X_1 | AND | f | XOR |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 0 |

- 뉴런에 입력되는 가중치의 합이 임계치를 초과하면 **1**, 아니면 **0**
- **AND**

$$0 \times W_0 + 0 \times W_1 = 0 < 0.5$$

$$0 \times W_0 + 1 \times W_1 = W_1 < 0.5$$

$$1 \times W_0 + 0 \times W_1 = W_0 < 0.5$$

$$1 \times W_0 + 1 \times W_1 = W_0 + W_1 > 0.5$$

→ W_0, W_1 : **0.3 or 0.4**

- **XOR의 경우**

$$0 \times W_0 + 0 \times W_1 = 0 < 0.5$$

$$0 \times W_0 + 1 \times W_1 = W_1 > 0.5$$

$$1 \times W_0 + 0 \times W_1 = W_0 > 0.5$$

$$1 \times W_0 + 1 \times W_1 = W_0 + W_1 < 0.5$$

→ 만족하는 W_0, W_1 는 존재하지 않음

→ 간단한 **XOR** 문제도 해결하지 못함

- 이러한 문제를 해결하기 위해서 **2개** 또는 **3개의 층(layer)**을 사용
- 다층 퍼셉트론: **3층** 퍼셉트론으로 어떤 문제도 해결가능
- 퍼셉트론은 다층 퍼셉트론 및 오류역전파 알고리즘의 기반 모델이 됨

다층 퍼셉트론 (Multi-Layer Perceptron)

- 입력층과 출력층 사이에 하나 이상의 은닉층을 가지는 전방향 신경회로망
- 단층 퍼셉트론의 문제점을 해결 → 효과적인 학습 알고리즘이 없어서 **60-70**년대 사용되지 않았다.
- **80년대** 중반 오류 역전파(**EBP**)알고리즘 학습규칙(**Rumelhart**)
→ 일반화된 델타 규칙(**generalized delta rule**)
- 알고리즘 : 원하는 목표값(**d**)과 실제 출력값(**o**) 사이의 오차제곱합으로 정의된 비용함수(**cost function**) **E**의 값을 경사하강추적법(**gradient-descent method**)에 의해 최소화 하는 방향으로 학습

$$E = \sum_p E_p, \quad (E_p = \frac{1}{2} \sum_j (d_{pj} - o_{pj})^2)$$

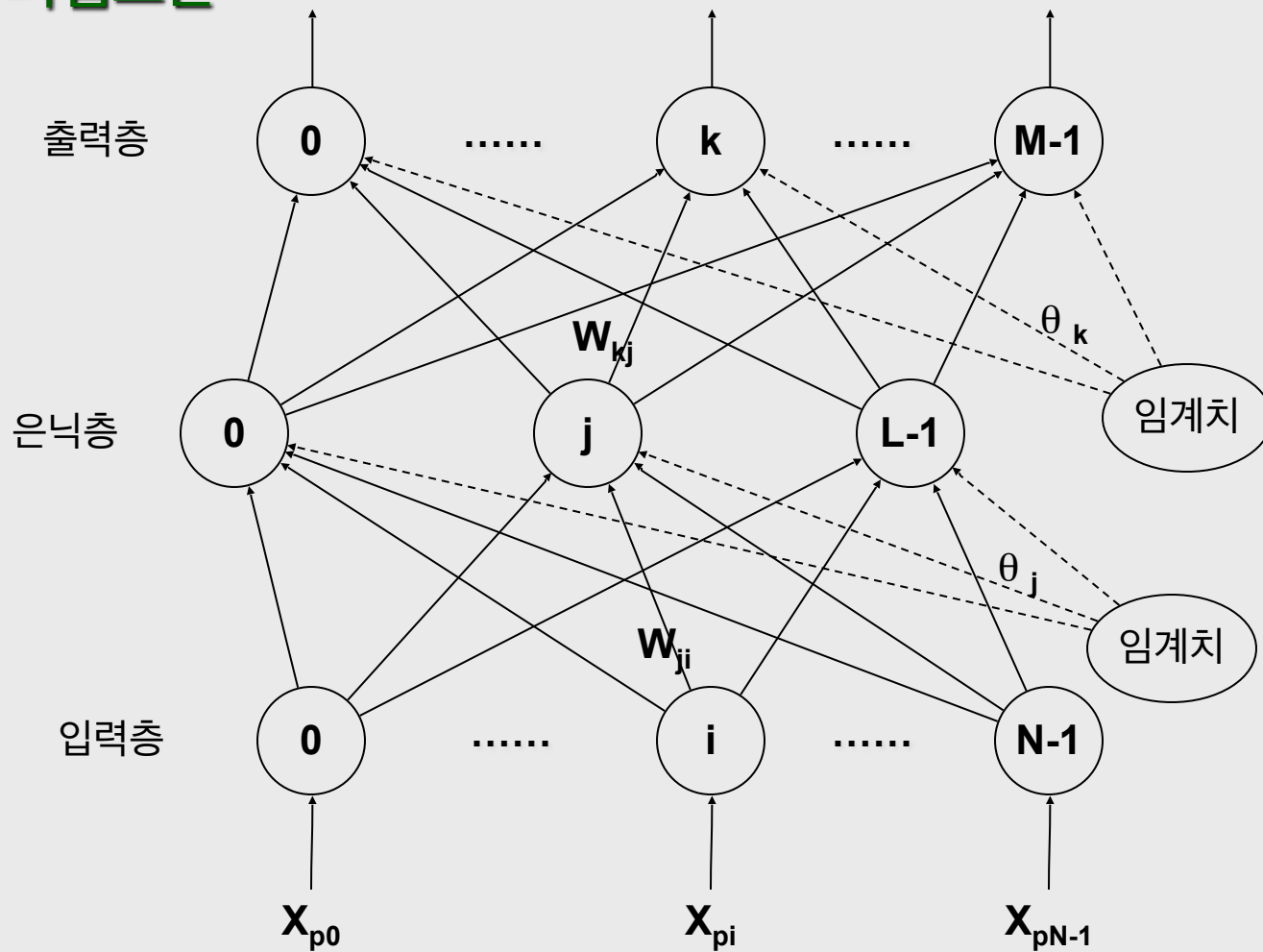
p: **p**번째 학습 패턴

E_p: **p**번째 패턴에 대한 오차

d_{pj}: **p**번째 패턴에 대한 목표출력의 **j**번째 요소

o_{pj}: **p**번째 패턴에 대한 실제 출력의 **j**번째 요소

● 2층 퍼셉트론



□ 학습 규칙

활성화 함수: 시그모이드 함수

1단계. 가중치와 임계치를 초기화

2단계. 입력과 목표 출력을 제시

3단계. 제시된 입력벡터를 이용하여 은닉층 **j**번째 뉴런으로 입력값 계산

$$net_{pj} = \sum_{i=0}^{N-1} W_{ji} X_{pi} - \theta_j$$

4단계. 시그모이드 함수를 사용하여 은닉층의 출력(**O_{pj}**)을 계산

5단계. 은닉층의 출력을 이용하여 출력층 뉴런 **k**로의 입력값을 계산

$$net_{pk} = \sum_{j=0}^{L-1} W_{kj} O_{pj} - \theta_k$$

6단계. 시그모이드 함수를 사용하여 출력층의 출력(**O_{pk}**)을 계산

7단계. 입력패턴의 목표출력(**d_{pk}**)과 실제 출력 (**O_{pk}**) 과의 오차값(**δ_{pk}**)을 계산하고
출력층 오차합(**E**)을 학습패턴의 오차(**E_p**)로 누적 시킨다.

$$\delta_{pk} = (d_{pk} - O_{pk}) f'_k(net_{pk}) = (d_{pk} - O_{pk}) O_{pk} (1 - O_{pk})$$

$$E = E + E_p, \quad (E_p = \sum_{k=1}^{M-1} \delta_{pk}^2)$$

8단계. 출력층 오차값(δ_{pk})과 은닉층과 출력층의 가중치값(W_{kj})을 이용하여 은닉층의 오차(δ_{pj})를 계산한다.

$$\delta_{pj} = f'_j(net_{pj}) \sum_{k=0}^{M-1} \delta_{pk} W_{kj} = \sum_{k=0}^{M-1} \delta_{pk} W_{kj} O_{pj} (1 - O_{pj})$$

9단계. 4단계와 7단계에서 구한 은닉층 뉴런 **j**의 출력값(O_{pj})과 출력층의 오차 값(δ_{pk})을 사용하여 출력층의 가중치(W_{kj})를 갱신한다(임계치도 조정)

$$W_{kj}(t+1) = W_{kj}(t) + \eta \delta_{pk} O_{pj}$$

$$\theta_k(t+1) = \theta_k(t) + \beta \delta_{pk}$$

10단계. 출력층에서와 마찬가지로 입력층과 은닉층의 가중치 값과 임계치값을 갱신한다.

$$W_{ji}(t+1) = W_{ji}(t) + \eta \delta_{pj} X_{pi}$$

$$\theta_j(t+1) = \theta_j(t) + \beta \delta_{pj}$$

11단계. 모든 학습패턴에 대하여 전부 학습 할 때까지 **2**단계로 분기하여 반복 수행한다.

12단계. 출력층의 오차합 **E**가 허용값 이하이거나 최대 반복회수보다 크면 종료, 그렇지 않으면 **2**단계로 가서 반복한다.

□ 오류역전파(error backpropagation) 개념

- 은닉층의 학습을 위해 출력층에서 발생한 오류를 이용하여 은닉층 오차계산
- 다시 이 값을 입력층으로 역전파시켜 출력층의 오차가 원하는 수준이 될 때까지 반복

□ 실제 적용시 고려사항

- 최소평균제곱 비용함수의 국부 최적해를 위해 뉴런수를 증가
- 이득함(η , β)을 낮출 필요 있음
- 초기 가중치를 다르게 하여 여러 번 학습하는 방법

□ 문제점

- 상위층의 목표값과 실제 출력값 간의 오류를 하위층으로 역전파시키는 것은 생물학적으로 타당하지 않음 (목표를 알지 못함)
- 오류 수렴까지 많은 회수의 학습 반복

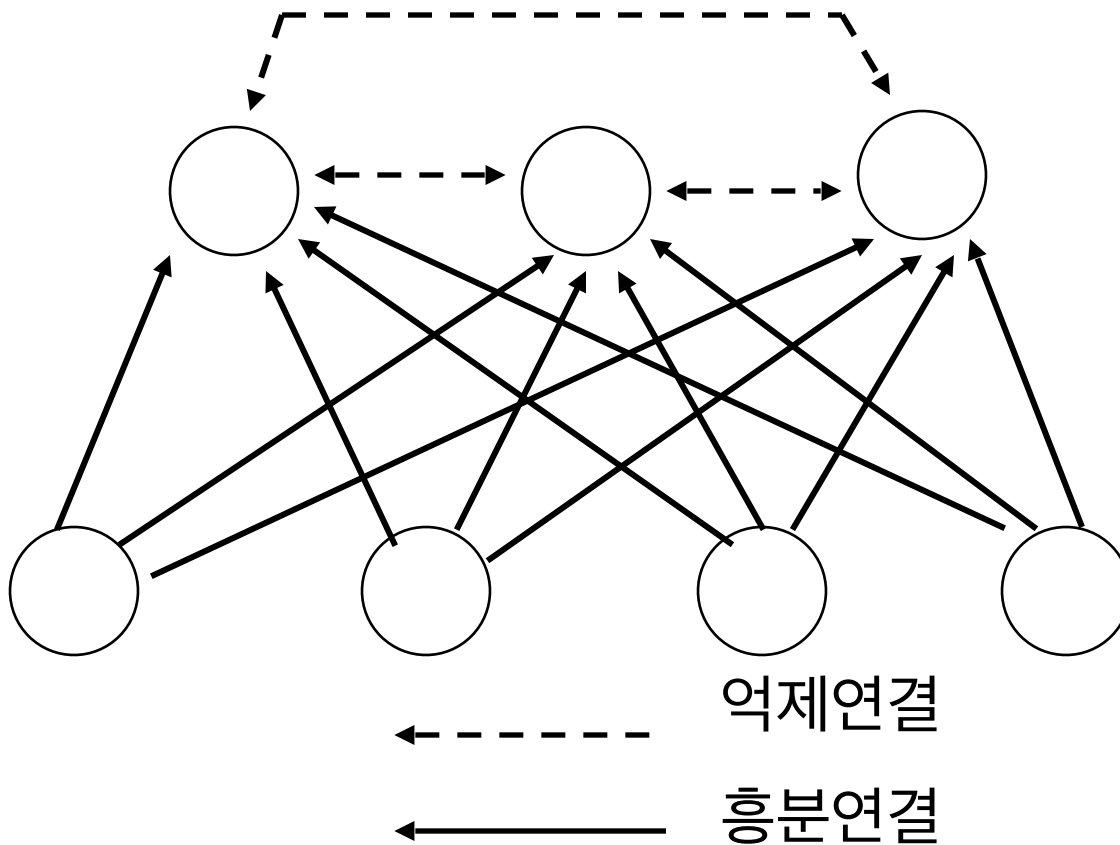
□ 지도학습에서 효율성 나타냄 → 보편적으로 많이 사용


경쟁학습 (Competitive Learning)

● 경쟁학습

- 알고리즘 : 입력벡터들을 신경회로망에 계속적으로 제시하면서 자율적으로 연결가중치를 변경시키는 방법
- 단순하면서 하드웨어 구현 시 구조 간단
 - 통계학의 **k-means** 군집화 알고리즘을 기초: 주어진 데이터를 **k**개의 클래스로 어느 오차수준 이하로 구분될 때까지 반복 → 패턴분류
- 단순 구조
 - 한 개의 입력층과 한 개의 출력층
 - 입력층과 출력층이 완전 연결
 - 입력층과 출력층 뉴런사이엔 흥분적으로 연결
 - 출력층의 뉴런들간에는 억제적으로 연결
 - 출력뉴런들은 승자 뉴런이 되기 위해 경쟁하고 오직 승자만이 학습함

단순 경쟁학습 신경회로망의 구조



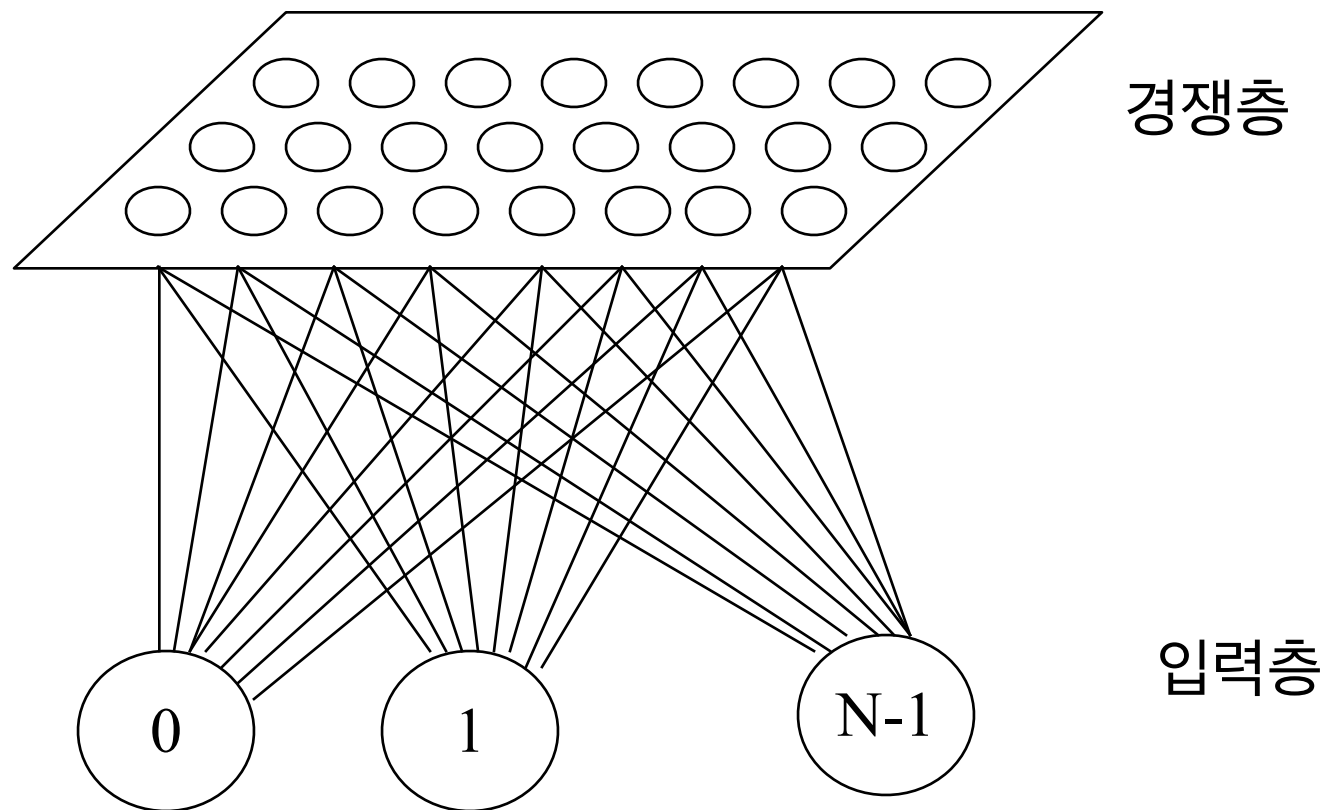


□ 초기 경쟁학습은 매번 승자뉴런만을 학습시키므로 초기 가중치벡터들의 분포에 따라 전혀 학습이 이루어지지 않은 출력뉴런들이 생기는 문제점

□ 자기조직화(self-organizing) 알고리즘

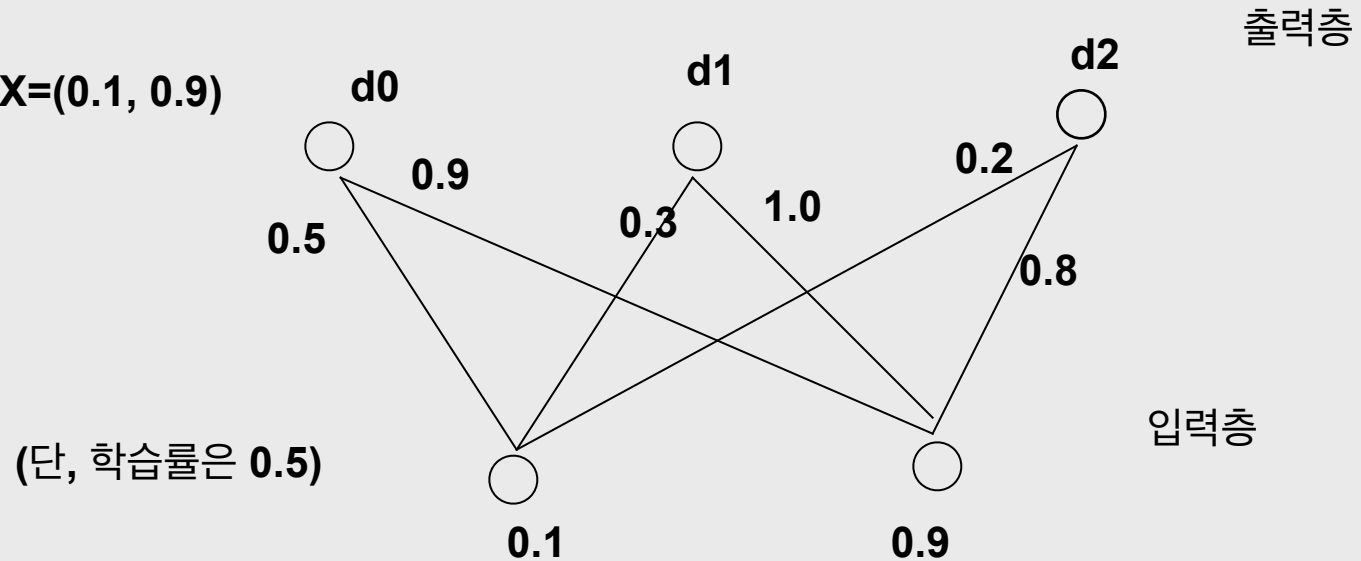
- 인접한 출력뉴런들은 비슷한 기능을 수행할 것이라는 예측(뇌의 위치에 따라 인지 종류가 다름)
- 기존의 경쟁학습을 개선하여 입력벡터와 가장 가까운 출력뉴런의 이웃뉴런들을 함께 학습시키는 알고리즘
- 승자뉴런뿐만 아니라 위상적으로 이웃한 뉴런들도 함께 학습
- 학습규칙
 - 1단계. 연결가중치를 초기화
 - 2단계. 새로운 입력패턴을 입력뉴런에 제시한다.
 - 3단계. 입력벡터와 모든 출력뉴런들과의 거리(입력-가중치)를 계산
 - 4단계. 최소거리를 가지는 승자뉴런을 구한다.
 - 5단계. 그 승자뉴런과 이웃한 출력뉴런에 연결된 가중치들을 갱신한다.
 - 6단계. 2단계로 가서 새로운 입력벡터를 처리한다.
 - 7단계. 지정된 학습회수까지 2단계부터 6단계의 과정을 반복

자기조직화 (Self-Organizing Feature Map) 신경회로망



□ 예제

입력: $X=(0.1, 0.9)$



- $d_0 = (0.1-0.5)^2 + (0.9-0.9)^2 = 0.16$, $d_1 = 0.05$, $d_2 = 0.02$ (승자뉴런)
- d_2 의 연결 가중치 조정

$$W_{02}(t+1) = 0.2 + 0.5(0.1 - 0.2) = 0.15$$

$$W_{12}(t+1) = 0.8 + 0.5(0.9 - 0.8) = 0.85$$

→ 제시된 입력과 가장 유사한 출력뉴런의 가중치벡터가 입력을 향하여 이동

Self-Organizing Feature Map (SOM) algorithm

[단계 1] 연결강도를 초기화한다.

N개의 입력으로부터 M개의 출력 뉴런 사이의 연결강도를 임의의 값으로 초기화
초기의 이웃반경은 아래 그림에서와 같이 모든 뉴런들이 포함될 수 있도록 충분히
크게 잡았다가 점차로 줄어든다.

[단계 2] 새로운 입력벡터를 제시한다.

[단계 3] 입력벡터와 모든 뉴런들간의 거리를 계산한다.

입력과 출력 뉴런 j사이의 거리 d_j 는 (식 2)와 같이 계산한다.

$$d_j = \sum_{i=0}^{N-1} (X_i(t) - W_{ij}(t))^2 \quad (\text{식 2})$$

여기서 $x_i(t)$ 는 시각 t에서의 i번째 입력벡터이고,
 $w_{ij}(t)$ 는 시각 t에서의 i번째 입력벡터와 j번째 출력 뉴런 사이의 연결강도이다.

[단계 4] 최소 거리에 있는 출력 뉴런을 선택한다.

최소 거리 d_j 인 출력 뉴런 j^* 를 선택한다.

[단계 5] 뉴런 j^* 와 그 이웃들의 연결강도를 재조정한다.

뉴런 j^* 와 그 이웃 반경내의 뉴런들의 연결강도를 다음식에 의해 재조정한다.

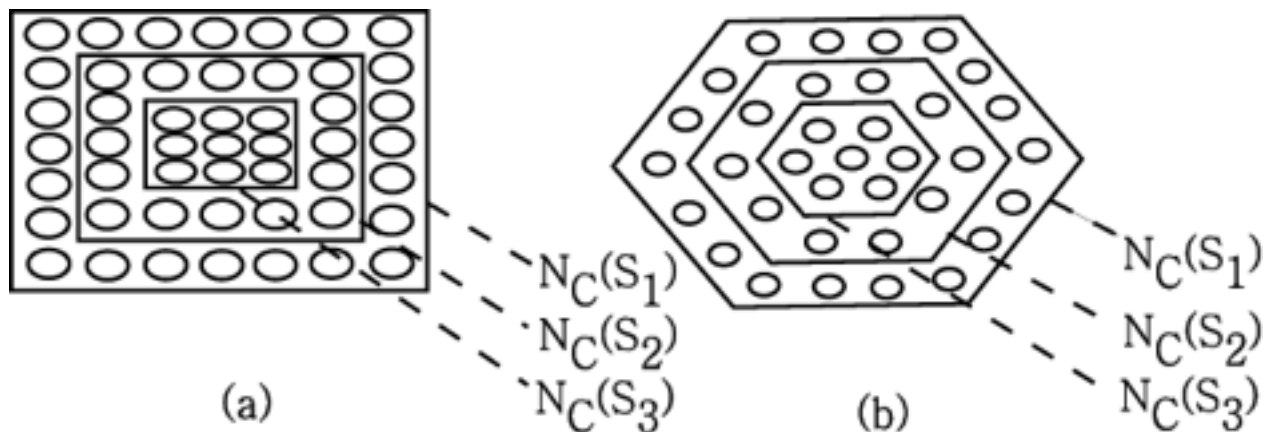
$$w_{ij}(t+1) = w_{ij}(t) + \alpha(x_i(t) - w_{ji}(t)) \quad (\text{식 3})$$

여기서 j 는 j^* 의 이웃 반경내의 뉴런이고,

i 는 0에서 $N-1$ 까지의 정수값이다.

α 는 0과 1사이의 값을 가지는 이득항(gain term)인데 시간이 경과함에 따라 점차 작아진다.

[단계 6] 단계 2로 가서 반복한다.



이웃반경의 크기 조정

ART 모델

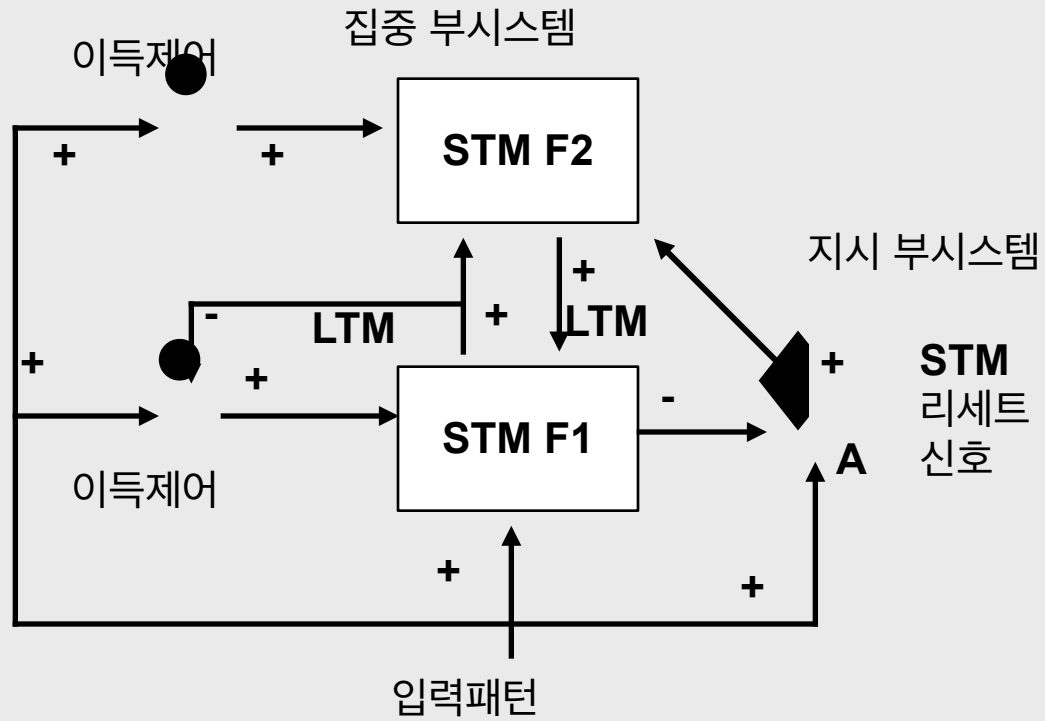
□ 적응 공명 이론(Adaptive Resonance Theory)

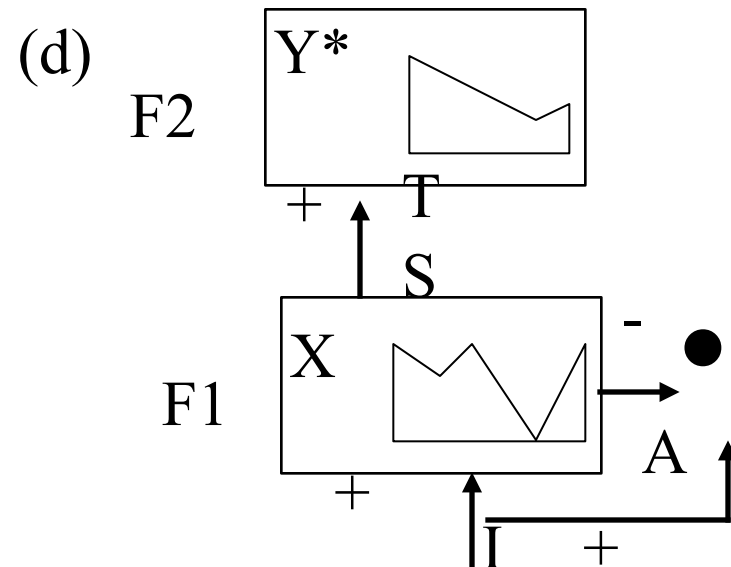
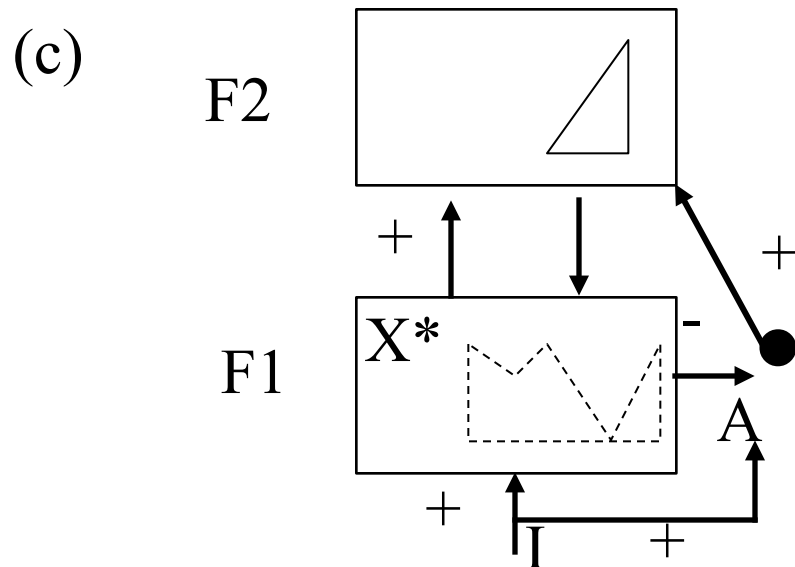
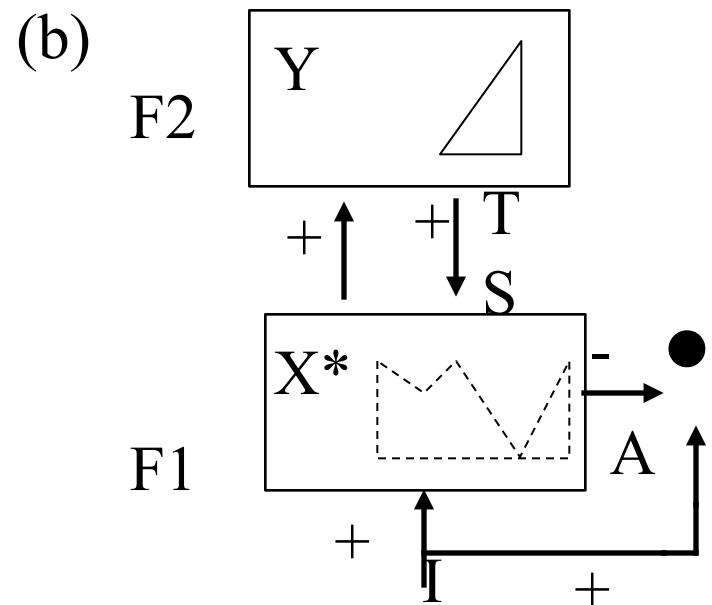
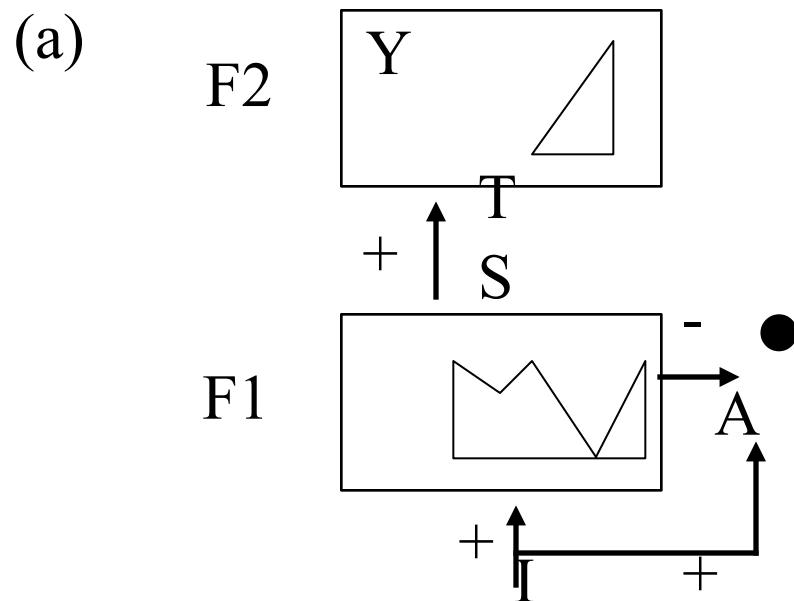
- 안정-적응(**stability-plasticity**)에 의한 자율 정규화
 - 중요한 사건에 대해서 이를 내부적으로 잘 표현(적응)
 - 중요치 않은 사건은 현재 학습된 내용에 손상을 주지않고 안정된(**stable**) 상태로 남는다
- 입력이 들어오면 기존의 학습된 지식과 유사한 지를 비교
 - 만일 비슷하면 새로운 입력을 기존의 학습된 내용과 결합시켜 학습된 지식을 수정하고,
 - 다르다면 입력패턴을 새로운 지식으로 받아들여서 기억장소에 저장하게 됨.

□ 구성

- 집중 부시스템(**attentional subsystem**): 이전에 학습된 사실과 비슷한 입력이 주어질 경우, 이에 대해 안정성있게 대응
- 지시 부시스템(**orienting subsystem**): 주어진 입력이 학습되어 있는 것인지 아닌지를 판별 → 아니면 집중 부시스템에 알려 새로운 사실 학습토록 함
- 이들 두 시스템에 의해 안정-적응 조절

□ ART 시스템 구성도







□ 학습 규칙

1단계. 신경회로망의 학습 요소들을 초기화

2단계. 새로운 입력을 제시

3단계. 매칭 점수를 계산

4단계. 입력패턴과 가장 잘 매칭이 되는 대표벡터를 선정

5단계. 유사도를 검사 → 유사도가 경계치를 넘으면 **7단계**, 아니면 **6단계**

6단계. 최상으로 매칭된 대표벡터를 억제 시킨 후 **3단계**로 분기

7단계. 최상으로 매칭된 대표벡터를 적응

8단계. **6단계**에서 억제 시킨 뉴런들을 모두 복원시킨 후에 **2단계**로 분기하여 반복 수행

□ 임의의 순서의 패턴에 대해 실시간으로 인식코드를 자율 구성하는 특성

→ 인간의 학습구조를 가장 잘 모방

□ 구조 복잡 → 실제 문제에 적용이 어렵다.

신경회로망 응용

- 화상처리, 문자인식, 음성인식 등의 패턴인식에 많이 응용
- 패턴인식: 미리 알고 있는 사실과 자료에 따라 미지의 입력을 클래스별로 분류하는 기술
- 연상기억: 일부 유실 되거나 변형이 가하여진 자료로부터 완전한 정보를 추출하는 기술
- 최적화 문제: 기존의 디지털 컴퓨터로는 다차 함수 시간 안에 풀 수 없는 **NP**문제이다. 알고리즘적 한계가 있으므로 **Hopfield** 신경회로망이나 **Kohonen** 신경회로망 등을 사용하여 **TSP**문제, 그래프 색칠하기, 채널 경로배정 문제를 해결
- 최근에는 정보여과, 데이터 마이닝 등에 응용