개념정리 #2

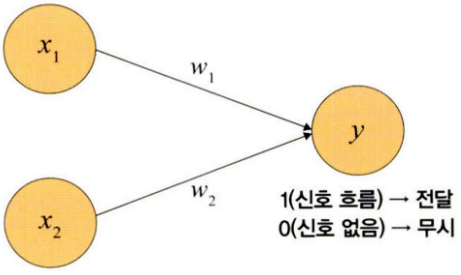
딥러닝 파이토치 교과서 4장

4. 딥러닝 시작

4.1 인공 신경망의 한계와 딥러닝 출현

퍼셉트론(perceptron) :

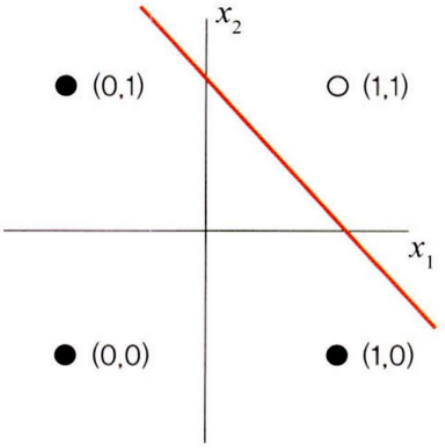
- 프랭크 로젠블라트(Frank Rosenblatt)가 1957년에 고안한 선형 분류기
- 입력층, 가중치, 출력층으로 구성된 구조
- 오늘날 신경망(딥러닝)의 기원이 되는 알고리즘
- 흐름이 있는 다수의 신호를 입력으로 받아 하나의 신호(1 또는 0)를 출력



AND 게이트

모든 입력이 '1'이어야 '1' 출력

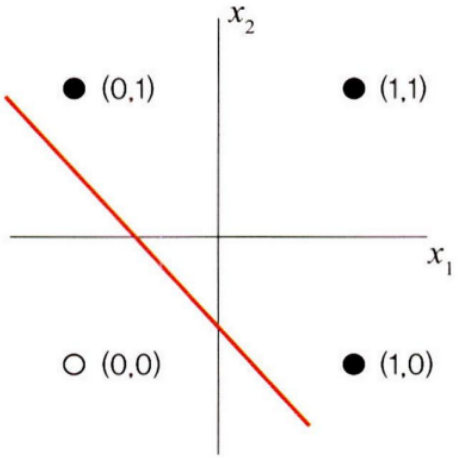
x_1	x_2	y
0	0	0
1	0	0
0	1	0
1	1	1



OR 게이트

둘 중 하나라도 '1'이면 '1' 출력

x_1	x_2	y
0	0	0
1	0	1
0	1	1
1	1	1

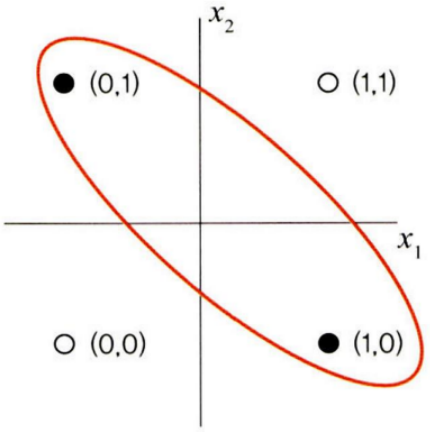


XOR 게이트

(배타적 논리합)

둘 중 한 개만 '1'일 때 '1' 출력

x_1	x_2	y
0	0	0
1	0	1
0	1	1
1	1	0



→ XOR 게이트는 데이터가 비선형적으로 분리되기 때문에 단층 퍼셉트론에서는 학습 불가능함

📌 **다층 퍼셉트론**(multi-layer perceptron) : 입력층과 출력층 사이에 하나 이상의 **은닉층**(hidden layer)을 두어 비선형적으로 분리되는 데이터에 대해서도 학습 가능

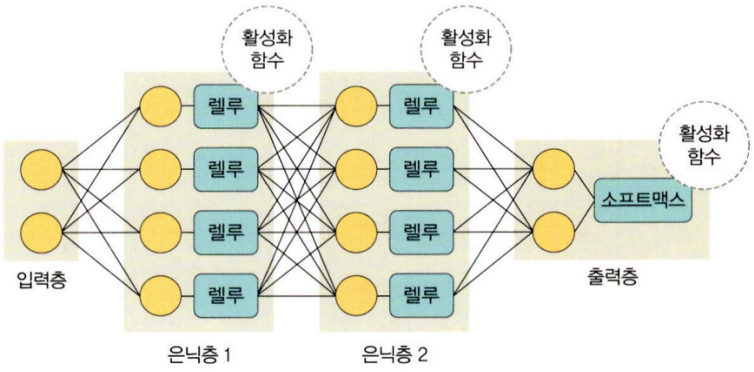
딥러닝

심층 신경망(Deep Neural Network, DNN)

: 입력층과 출력층 사이에 은닉층이 여러 개 있는 신경망

4.2 딥러닝 구조

딥러닝 용어



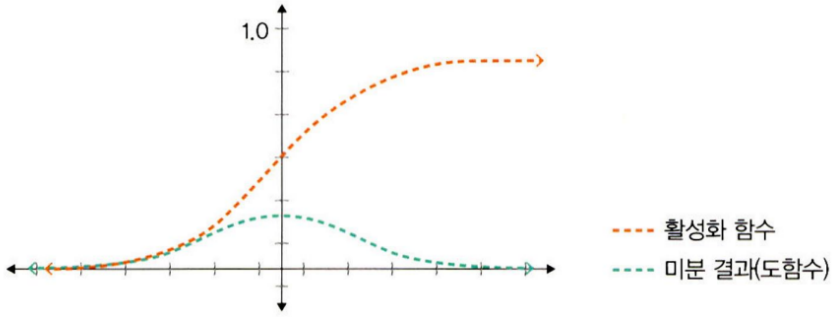
입력층(input layer)	데이터를 받아들이는 층
은닉층(hidden layer)	모든 입력 노드로부터 값을 받아 가중합을 계산하고, 이 값을 활성화 함수에 적용하여 출력층에 전달하는 층
출력층(output layer)	신경망의 최종 결괏값이 포함된 층
가중치(weight)	노드와 노드 간 연결 강도
바이어스(bias)	가중합에 더해주는 상수. 최종적으로 출력되는 값을 조절하는 역할을 함
가중합(weighted sum), **전달 함수**	가중치와 신호의 곱을 합한 것
활성화 함수(activation function)	신호를 입력받아 이를 적절히 처리하여 출력해 주는 함수
손실 함수(loss function)	가중치 학습을 위해 출력 함수의 결과와 실젯값 간의 오차를 측정하는 함수

가중치 : 입력 값이 연산 결과에 미치는 영향력을 조절하는 역할 (e.g. 가중치 w_1이 0.001이라면 입력 값 x_1이 아무리 큰 값이어도 w_1x_1은 0에 가까운 값이 됨)

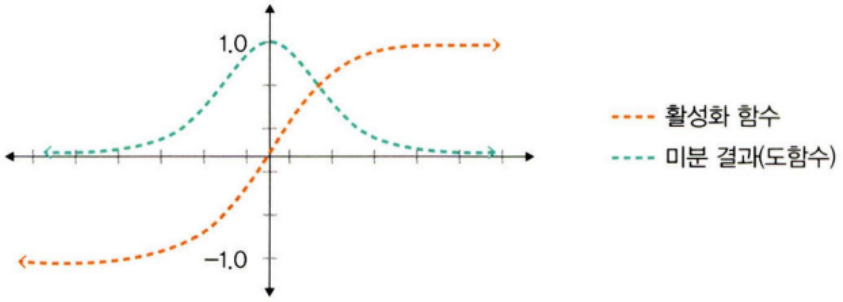
가중합 (전달 함수) : $\Sigma w_i x_i + b$

활성화 함수 :

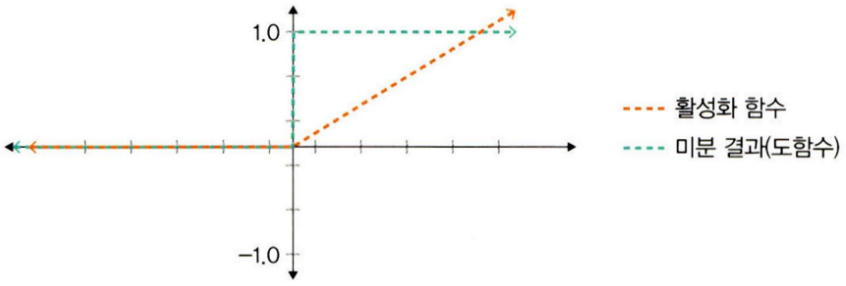
- **Sigmoid 함수**
  - $f(x) = \frac{1}{1+e^{-x}}$
  - 선형 함수의 결과를 0~1 사이의 비선형 형태로 변형
  - Vanishing gradient problem 有



- **Tanh 함수**
  - 하이퍼볼릭 탄젠트 함수
  - 선형 함수의 결과를 -1~1 사이의 비선형 형태로 변형
  - Zero-centered
  - Vanishing gradient problem 有

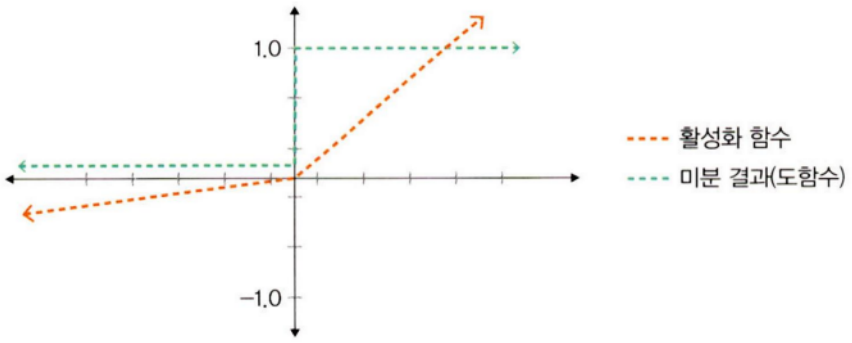


- **ReLU 함수**
  - 입력이 양수이면 1 출력, 음수이면 0 출력
  - 경사 하강법에 영향을 주지 않아 학습 속도가 빠름
  - 기울기 소멸 문제 X
  - 음수 입력받았을 때 0을 출력하므로 학습 능력 감소



- **Leaky ReLU 함수**

- 입력이 양수이면 1 출력, 음수이면 매우 작은 수 반환



- **Softmax 함수**
  - $y_k = \frac{\exp(a_k)}{\Sigma_{i=1}^{n}\exp(a_i)}$
  - 출력값들의 총합이 항상 1이 되도록 (확률)

손실 함수 :

- c.f. 경사 하강법(gradient descent) : 학습률(η, learning rate)과 **손실 함수**의 순간 기울기(그래디언트)를 이용하여 가중치를 업데이트하는 방법. 오차를 비교하고 최소화하는 방향으로 이동.
- 손실 함수는 학습을 통해 얻은 데이터의 추정치(예측값)가 실제 데이터(정답)와 얼마나 차이가 나는지 평가하는 지표
- **평균 제곱 오차**(**M**ean **S**quared **E**rror)
  - $MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$
  - 실젯값과 예측값의 차이의 제곱의 평균
- **크로스 엔트로피 오차**(Cross Entropy Error)
  - $CrossEntropy = \sum_{i=1}^{n} y_i \log \hat{y}_i$
  - Classification 문제에서, one-hot encoding을 했을 때만 사용할 수 있음

    — Classification 문제 :

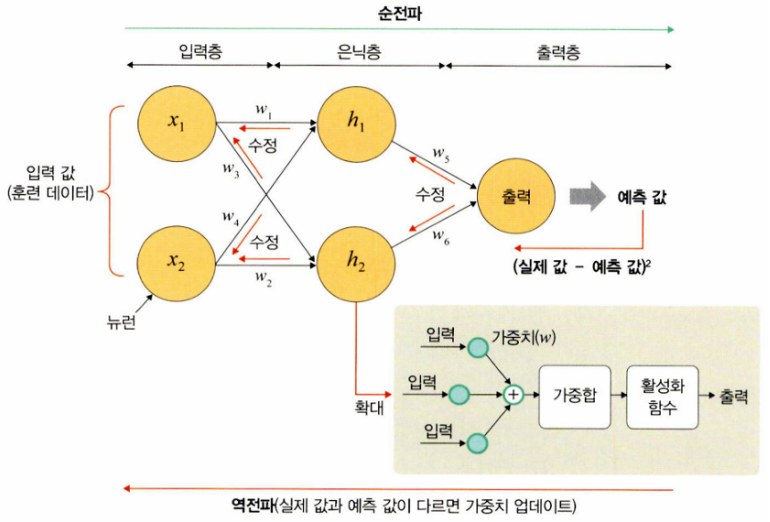
    데이터의 출력을 0, 1로 구분하기 위해 시그모이드 함수 사용

    → 시그모이드 함수의 자연 상수 e 때문에 MSE를 적용하면 울퉁불퉁한 그래프가 출력됨 (MSE는 출력값이 실젯값과 얼마나 다른지를 제곱하여 계산. 이때 시그모이드 함수의 출력이 0.5 근처일 때 손실 변화가 급격히 일어나고, 0 또는 1

에 가까울 때는 변화가 완만해짐. 이로 인해 손실 함수의 그래프가 울퉁불퉁하게 보일 수 있음.)

→ 크로스 엔트로피 손실 함수 사용 : Local minima 문제 방지하기 위해 자연로그를 모델의 출력값에 취함 (크로스 엔트로피 손실 함수는 Local minima의 영향을 덜 받도록 설계됨. 모델의 예측이 잘못된 경우 손실이 급격히 증가하여 기울기가 더 강하게 작용.)

딥러닝 학습



순전파(feedforward) : 네트워크에 training data가 들어올 때 발생. 모든 뉴런이 이전 층의 뉴런으로부터 수신한 정보에 가중합 및 활성화 함수를 적용해 다음 층의 뉴런으로 전송함.

손실 함수(loss function)로 예측값과 실젯값의 차이인 loss를 추정함. (→ loss가 0에 가까울수록 좋은 것)

∴ 손실 함수 비용(cost)이 0에 가까워지도록 모델이 훈련을 반복하면서 가중치를 조정함

Loss가 계산되면 그 정보가 역으로(output → hidden → input layer) 전파됨 : **역전파** (backpropagation)

⇒ 여기서 계산된 각 뉴런 결과를 또다시 순전파의 가중치 값으로 사용함

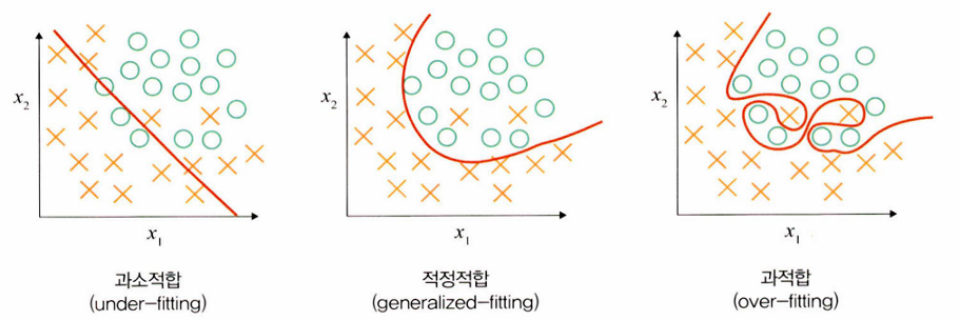
딥러닝의 문제점과 해결 방안

활성화 함수가 적용된 은닉층 개수가 많을수록 → 비선형적으로 표현 잘됨 → 데이터 분류 잘됨

BUT

1. **Overfitting Problem**

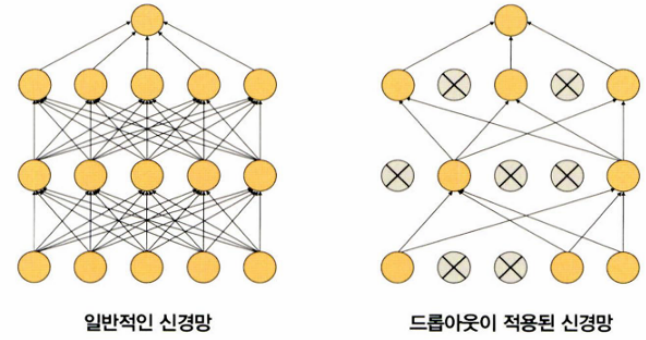
   : Training data를 과도하게 학습하여 test data 및 실제 데이터에 대해 오히려 오차가 증가하는 현상



   해결 방안

   - **Dropout**

     : 학습 과정 중에 임의로 일부 노드들을 학습에서 제외시킴



2. **Vanishing gradient problem**

   : 출력층에서 은닉층으로 전달되는 loss가 크게 줄어들어서 학습되는 양이 '0'에 가까워져 학습이 잘 되지 않는 현상 (가중치 업데이트가 거의 되지 않음). Hidden layer가 많은 신경망에서 주로 발생함.

   해결 방안

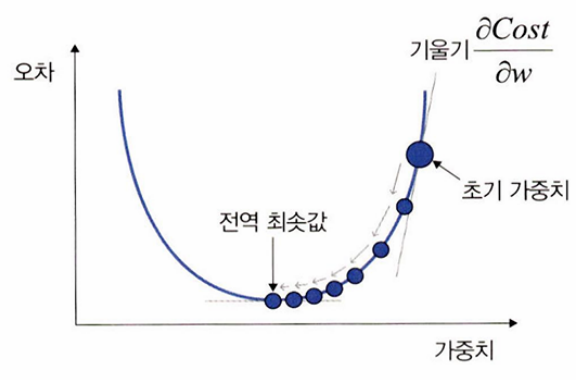
   - **활성화 함수 -** 시그모이드/tanh 대신 **ReLU 함수** 사용

     (시그모이드, tanh 함수: x가 0에서 멀어질수록 기울기가 0에 수렴(saturated)

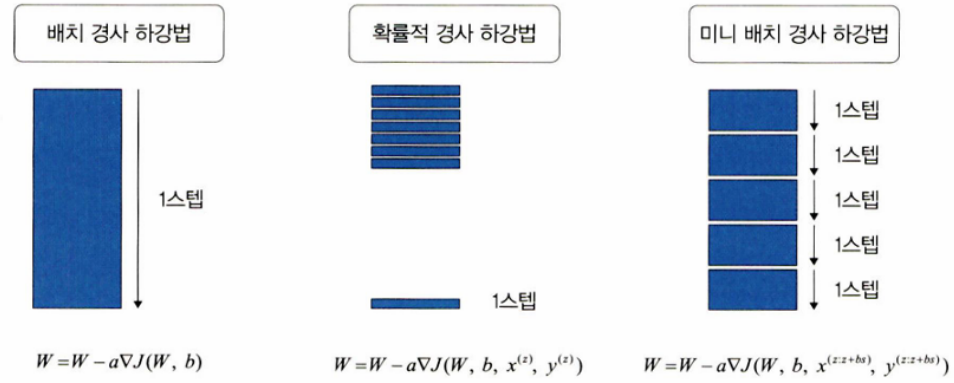
but ReLU 함수 : x≥0일 때 기울기 계속 증가)

3. **성능이 나빠지는 문제**

: 경사 하강법(gradient descent) - loss function의 비용이 최소가 되는 지점을 찾을 때까지 기울기가 낮은 쪽으로 계속 이동시키는 과정을 반복함. 이때 성능이 나빠지는 문제 발생



해결 방안



- **배치 경사 하강법**(Batch Gradient Descent, BGD)

  : 전체 데이터셋에 대한 loss를 구한 후, 기울기를 한 번만 계산해 모델의 파라미터를 업데이트 → 전체 training dataset에 대해 가중치를 편미분

  : $W = W - \alpha\nabla J(W, b)$

- **확률적 경사 하강법**(Stochastic Gradient Descent, SGD)

  : 임의로 선택한 데이터에 대해 기울기를 계산 ~ 적은 데이터를 사용하므로 빠른 계산 가능

- **미니 배치 경사 하강법**(mini-batch gradient descent)

: 전체 데이터셋을 여러 개의 미니 배치로 나누고, 미니 배치 한 개마다 기울기를 구한 후 그것들의 평균 기울기를 이용해 모델을 업데이트

~ SGD보다 안정적이고 BGD보다 빠름

> 💡 **Optimizer**
>
> 학습 속도와 운동량을 조정하는 옵티마이저 적용 가능

딥러닝을 사용할 때 이점

1. **특성 추출 (**feature extraction)

   컴퓨터가 입력받은 데이터를 분석해 일정한 패턴이나 규칙을 찾아내려면 사람이 인지하는 데이터를 컴퓨터가 인지할 수 있는 데이터로 변환해 주어야 함 → 데이터별 특징을 찾아내고 그것을 토대로 데이터를 벡터로 변환하는 작업이 특성 추출

   ⇒ 딥러닝에서는 데이터 특성을 잘 잡아내고자 hidden layer를 깊게 쌓는 방식으로 파라미터를 늘려 특성 추출 과정을 알고리즘에 통합시킴.

2. **빅데이터의 효율적 활용**

   딥러닝 학습을 이용한 특성 추출은 데이터 샘플이 많을수록 성능이 향상됨 → 빅데이터 이용

4.3 딥러닝 알고리즘

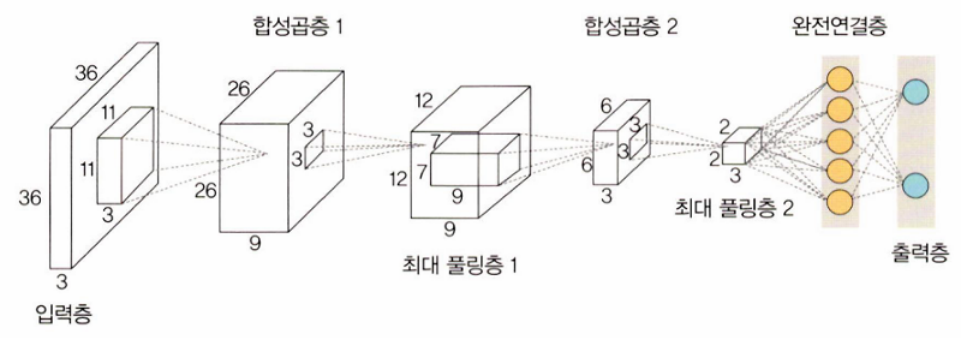
심층 신경망 (Deep Neural Network, DNN)

: Input layer와 output layer 사이에 다수의 hidden layer를 포함하는 인공 신경망

- 장점 : 다수의 hidden layer → 다양한 비선형적 관계 학습 가능
- 단점 : 학습을 위한 연산량 많음, 기울기 소멸 문제 발생 가능

  ⇒ Dropout, ReLU 함수, batch normalization 적용해야 함

합성곱 신경망 (Convolutional Neural Network, CNN)

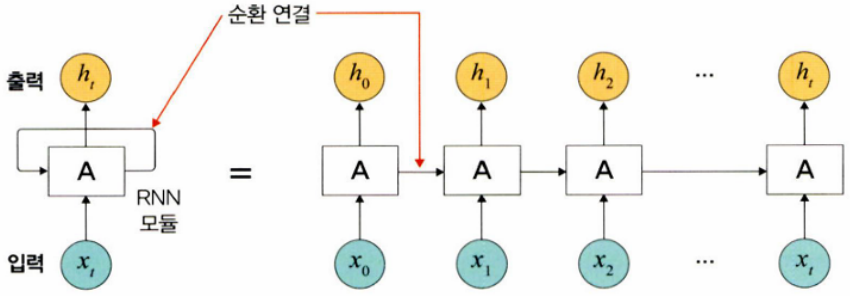


: 이미지에서 객체, 얼굴, 장면 등을 인식하기 위해 패턴을 찾는 데에 특히 유용함

- 각 층의 입출력 형상을 유지함
- 이미지의 공간 정보를 유지하면서 인접 이미지와 차이가 있는 특징을 효과적으로 인식함
- 복수 필터로 이미지의 특징을 추출 및 학습함
- Pooling layer : 추출한 이미지의 특징을 모으고 강화함
- 필터를 공유 파라미터로 사용 - 학습 파라미터가 매우 적음

순환 신경망 (Recurrent Neural Network, RNN)

: 시계열 데이터, 시간 흐름에 따라 변화하는 데이터를 학습하기 위한 인공 신경망



- 순환(recurrent) : '자기 자신을 참조한다' - 현재 결과가 이전 결과와 연관
- 시간성을 가진 데이터가 많음
- 시간성 정보를 이용해 데이터의 특징을 잘 다룸
- 시간에 따라 내용이 변하므로 데이터가 동적, 길이가 가변적
- 매우 긴 데이터를 처리하는 연구 활발히 진행 중
- Vanising gradient problem 有 → LSTM(Long-Short Term Memory) 사용

제한된 볼츠만 머신 (Restricted Boltzmann Machine, RBM)

볼츠만 머신 : Visible layer와 hidden layer로 구성된 모델

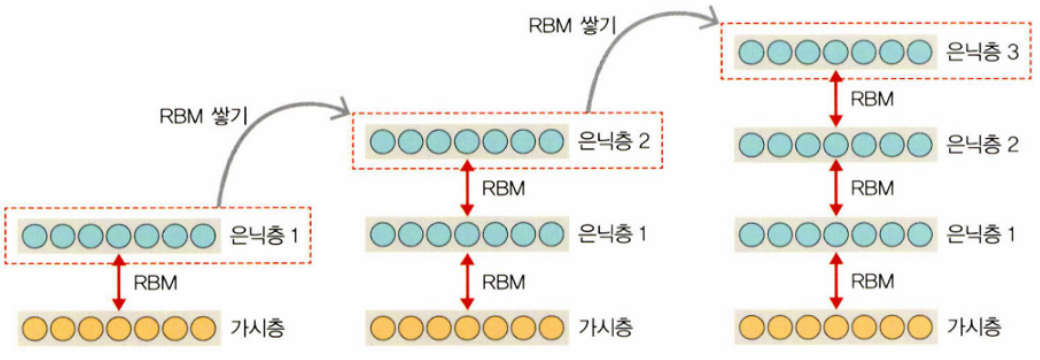
제한된 볼츠만 머신 : Visible layer 간에, hidden layer 간에 연결이 없는 모델 (visible layer와 hidden layer 사이의 연결만 있음)

- Dimensionality reduction, classification, linear regression, collaborative filtering, feature learning, topic modelling에 사용
- Vanishing gradient problem 해결 위해 사전 학습 용도로 활용 가능
- 심층 신뢰 신경망(DBN)의 요소로 활용됨

심층 신뢰 신경망 (Deep Belief Network, DBN)

: 사전 훈련된 RBM을 블록처럼 여러 층으로 쌓은 형태로 연결된 신경망

- 레이블이 없는 데이터에 대한 비지도 학습 가능
- 부분적인 이미지에서 전체를 연상하는 일반화, 추상화 과정을 구현할 때 유용함
- 학습 절차
  1. Visible layer와 Hidden layer 1에 RBM을 사전 훈련
  2. 첫 번째 층 input data와 파라미터를 고정, 두 번째 층 RBM을 사전 훈련
  3. 원하는 층 개수만큼 RBM을 쌓아 올려서 전체 DBN 완성



- 순차적으로 DBN을 학습시키며 계층적 구조 생성
- 비지도 학습
- 위로 올라갈수록 추상적 특성 추출
- 학습된 가중치를 다층 퍼셉트론의 가중치 초깃값으로 사용함