

벡터의 유사도



- 벡터의 유사도
Robot Media Laboratory

- 문장이나 문서의 유사도



● 벡터의 유사도

Robot Media Laboratory

● 문장이나 문서의 유사도

- 문서의 유사도는 주로 문서들 간에 동일한 단어 또는 비슷한 단어가 얼마나 공통적으로 많이 사용되었는지에 의존



● 벡터의 유사도

Robot Media Laboratory

● 벡터의 유사도

- 코사인 유사도
- 자카드 유사도
- 유클리디안 유사도
- 맨하탄 유사도

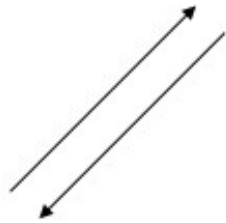


● 벡터의 유사도

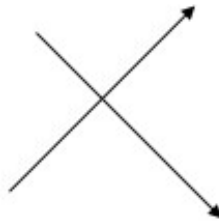
Robot Media Laboratory

● 코사인 유사도(Cosine Similarity)

- 두 벡터 간의 코사인 각도를 이용하여 구할 수 있는 두 벡터의 유사도



코사인 유사도 : -1



코사인 유사도 : 0



코사인 유사도 : 1

$$\text{similarity} = \cos(\Theta) = \frac{A \cdot B}{\|A\| \|B\|} = \frac{\sum_{i=1}^n A_i \times B_i}{\sqrt{\sum_{i=1}^n (A_i)^2} \times \sqrt{\sum_{i=1}^n (B_i)^2}}$$



● 벡터의 유사도

Robot Media Laboratory

● 자카드 유사도

- 합집합에서 교집합의 비율을 바탕으로 대상의 유사도 계산방식
- 0과 1사이의 값 동일하다면 1, 공통 원소가 없다면 0

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$



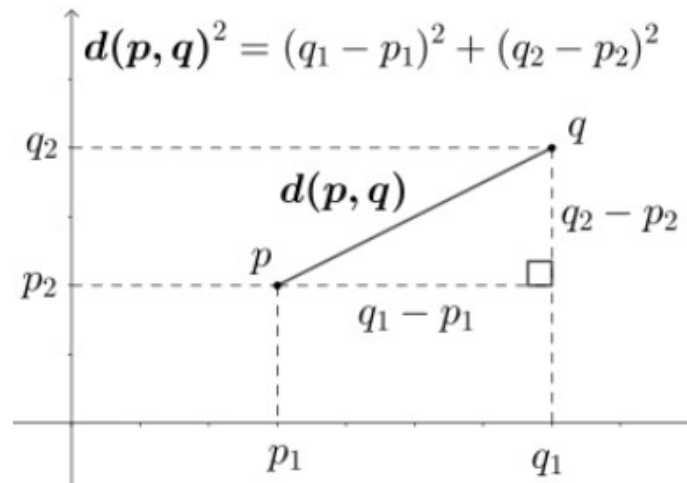
● 벡터의 유사도

Robot Media Laboratory

● 유클리디안(유클리드 거리) 유사도

- 피타고라스의 정리를 통해 두 점 사이의 거리를 구하는 것과 동일
- 점과 점사이의 거리

$$\sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \dots + (q_n - p_n)^2} = \sqrt{\sum_{i=1}^n (q_i - p_i)^2}$$



● 벡터의 유사도

Robot Media Laboratory

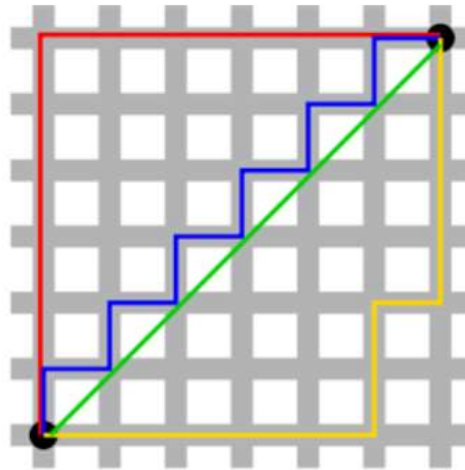
● 맨하탄 유사도

- 맨하탄거리를 이용하여 유사도 측정

$$MaDistance = \sum_{i=1}^n |a_i - b_i|$$

● 맨하탄거리

- 격자로 이뤄진 공간에서 출발점에서 도착점까지 길을 따라 만들어진 최단거리



● 벡터의 유사도

Robot Media Laboratory

방법	특징	수식
자카드 유사도	<ul style="list-style-type: none"> 두 문장을 각 단어의 집합으로 만든 뒤, 집합을 통해 유사도를 측정 1에 가까울수록 유사도가 높음 0~1사이의 값을 가짐 	$J(A, B) = \frac{ A \cap B }{ A \cup B } = \frac{ \text{token in } A \cap \text{token in } B }{ \text{token in } A \cup \text{token in } B }$
유클리디언 유사도	<ul style="list-style-type: none"> 거리를 측정하는 공식과 같음, L_2거리라고도 함 n차원 공간에서 두 점 사이의 최단 거리를 구하는 접근법 1보다 큰 값이 나올 수 있기 때문에 값의 제한을 걸어줌(L_1정규화) 	$d(x, y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \dots + (x_n - y_n)^2}$
맨하탄 유사도	<ul style="list-style-type: none"> 사각형 격자로 이루어진 지도에서 출발점에서 도착점까지 가로지르지 않고 갈 수 있는 최단거리, L_1거리라고도 함 	$MaDistance = \sum_{i=1}^n a_i - b_i $
코사인 유사도	<ul style="list-style-type: none"> 두 벡터 사이의 코사인 각도를 구하는 방법 -1~1사이를 가지고 1에 가까울수록 유사함 가장 널리 쓰임, 다른 방법보다 유사도가 정확함 → 단순 좌표상의 거리 가 아닌, 단어의 공간에서 벡터 간의 각도를 구하기 때문에 	$\cos(\theta) = \frac{\vec{a} \cdot \vec{b}}{ \vec{a} \cdot \vec{b} }$



탐색적 데이터 분석



● EDA(탐색적 데이터 분석)

- 수집한 데이터가 들어왔을 때, 이를 다양한 각도에서 관찰하고 이해하는 과정
- 데이터를 분석하기 전에 그래프나 통계적인 방법으로 자료를 직관적으로 바라보는 과정
- 데이터의 분포 및 값을 검토함으로써 데이터가 표현하는 현상을 더 잘 이해하고, 데이터에 대한 잠재적인 문제를 발견 가능
- 이를 통해, 본격적인 분석에 들어가기에 앞서 데이터의 수집을 결정 가능
- 다양한 각도에서 살펴보는 과정을 통해 문제 정의 단계에서 미처 발생하지 못했을 다양한 패턴을 발견하고, 이를 바탕으로 기존의 가설을 수정하거나 새로운 가설을 세울 수 있다.



- **분석의 목적과 변수가 무엇이 있는지 확인.**
 - 개별 변수의 이름이나 설명을 가지는지 확인
- **데이터를 전체적으로 살펴보기**
 - 데이터에 문제가 없는지 확인.
 - head나 tail부분을 확인, 추가적으로 다양한 탐색(이상치, 결측치 등을 확인하는 과정)
- **데이터의 개별 속성값을 관찰**
 - 각 속성 값이 예측한 범위와 분포를 갖는지 확인. 만약 그렇지 않다면, 이유가 무엇인지를 확인.
 - 속성 간의 관계에 초점을 맞추어, 개별 속성 관찰에서 찾아내지 못했던 패턴을 발견 (상관관계, 시각화 등)



성능 최 적화



● 성능 최적화

Robot Media Laboratory

● 데이터를 사용한 성능 최적화

- 데이터를 사용한 성능 최적화 방법은 많은 데이터를 수집하는 것
- 데이터 수집이 여의치 않은 상황에서는 임의로 데이터를 생성하는 방법도 고려해 볼 수 있음

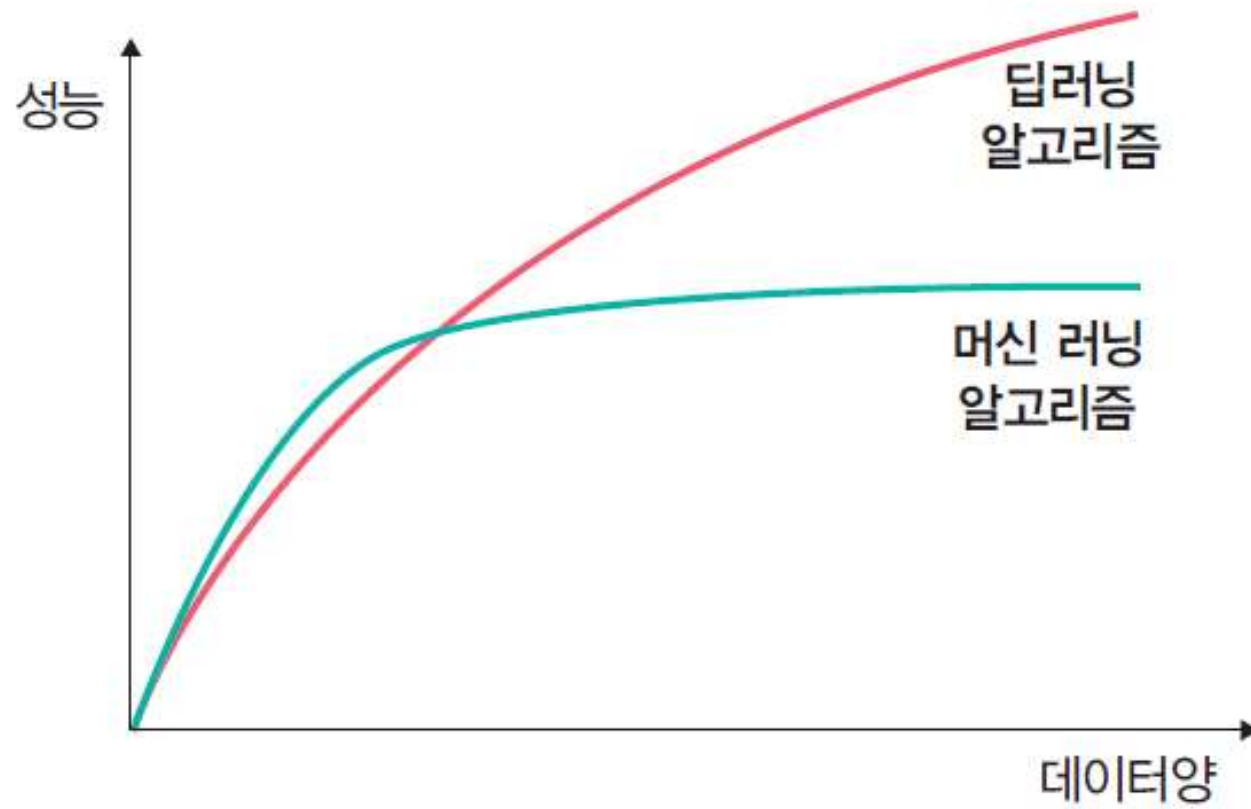
● 최대한 많은 데이터 수집하기

- 일반적으로 딥러닝이나 머신 러닝 알고리즘은 데이터양이 많을 수록 성능이 좋음
- 가능한 많은 데이터(빅데이터)를 수집해야 함



● 성능 최적화

Robot Media Laboratory



● 성능 최적화

Robot Media Laboratory

● 데이터 생성하기

- 많은 데이터를 수집할 수 없다면 데이터를 만들어 사용할 수 있음

● 데이터 범위(scale) 조정하기

- 활성화 함수로 시그모이드를 사용한다면 데이터셋 범위를 0~1의 값을 갖도록 하고,
- 하이퍼볼릭 탄젠트를 사용한다면 데이터셋 범위를 -1~1의 값을 갖도록 조정할 수 있음

● 정규화, 규제화, 표준화도 성능 향상에 도움이 됨



● 알고리즘을 이용한 성능 최적화

- 머신 러닝과 딥러닝을 위한 알고리즘은 상당히 많음
- 수많은 알고리즘 중 우리가 선택한 알고리즘이 최적의 알고리즘이 아닐 수도 있음
- 유사한 용도의 알고리즘들을 선택하여 모델을 훈련시켜 보고 최적의 성능을 보이는 알고리즘을 선택해야 함
- 머신 러닝에서는 데이터 분류를 위해 SVM, K-최근접 이웃 알고리즘들을 선택하여 훈련시켜 보거나,
- 시계열 데이터의 경우 RNN, LSTM, GRU 등의 알고리즘을 훈련시켜 성능이 가장 좋은 모델을 선택하여 사용



● 성능 최적화

Robot Media Laboratory

● 알고리즘 튜닝을 위한 성능 최적화

- 성능 최적화를 하는 데 가장 많은 시간이 소요되는 부분
- 모델을 하나 선택하여 훈련시키려면 다양한 하이퍼파라미터를 변경하면서 훈련시키고 최적의 성능을 도출해야 함

● 진단

- 성능 향상이 어느 순간 멈추었다면 원인을 분석할 필요가 있음
- 문제를 진단하는데 사용할 수 있는 것이 모델에 대한 평가
- 훈련(train) 성능이 검증(test)보다 눈에 띄게 좋다면 과적합을 의심
 - 해결하기 위해 규제화
- 훈련과 검증 결과가 모두 성능이 좋지 않다면 과소적합(under-fitting)을 의심
 - 과소적합 상황에서는 네트워크 구조를 변경하거나 훈련을 늘리기 위해 에포크 수를 조정
- 훈련 성능이 검증을 넘어서는 변곡점이 있다면 조기 종료를 고려



● 성능 최적화

Robot Media Laboratory

● 가중치

- 가중치에 대한 초기값은 작은 난수를 사용
- 작은 난수라는 숫자가 애매하다면 오토인코더 같은 비지도 학습을 이용하여 사전 훈련(가중치 정보를 얻기 위한 사전 훈련)을 진행한 후 지도 학습을 진행하는 것도 방법

● 학습률

- 학습률은 모델의 네트워크 구성에 따라 다르기 때문에 초기에 매우 크거나 작은 임의의 난수를 선택하여 학습 결과를 보고 조금씩 변경해야 함
 - 네트워크의 계층이 많다면 학습률은 높아야 하며, 네트워크의 계층이 몇 개 되지 않는다면 학습률은 작게 설정해야 함



● 성능 최적화

Robot Media Laboratory

● 활성화 함수

- 활성화 함수의 변경은 신중해야 함
- 활성화 함수를 변경할 때 손실 함수도 함께 변경해야 하는 경우가 많기 때문
- 일반적으로는 활성화 함수로 시그모이드나 하이퍼볼릭 탄젠트를 사용했다면 출력층에서는 소프트맥스나 시그모이드 함수를 많이 선택

● 배치와 에포크

- 일반적으로 큰 에포크와 작은 배치를 사용하는 것이 최근 딥러닝의 트렌드이기는 하지만, 적절한 배치 크기를 위해 훈련 데이터셋의 크기와 동일하게 하거나 하나의 배치로 훈련을 시켜 보는 등 다양한 테스트를 진행하는 것이 좋음



● 성능 최적화

Robot Media Laboratory

● 옵티마이저 및 손실 함수

- 일반적으로 옵티마이저는 확률적 경사 하강법을 많이 사용
- 네트워크 구성에 따라 차이는 있지만 아담(Adam)이나 알엠에스프롭(RMSProp) 등도 좋은 성능을 보이고 있음
- 다양한 옵티마이저와 손실 함수를 적용해 보고 성능이 최고인 것을 선택

● 네트워크 구성

- 네트워크 구성은 네트워크 토폴로지(topology)라고도 함
- 최적의 네트워크를 구성하는 것 역시 쉽게 알 수 있는 부분이 아니기 때문에 네트워크 구성을 변경해 가면서 성능을 테스트해야 함
- 하나의 은닉층에 뉴런을 여러 개 포함시키거나(네트워크가 넓다고 표현),
- 네트워크 계층을 늘리되 뉴런 개수는 줄여 봄(네트워크가 깊다고 표현)
- 혹은 두 가지를 결합하는 방법으로 최적의 네트워크가 무엇인지 확인한 후 사용할 네트워크를 결정해야 함



● 성능 최적화

Robot Media Laboratory

● 앙상블을 이용한 성능 최적화

- 앙상블은 간단히 모델을 두 개 이상 섞어서 사용하는 것
- 앙상블을 이용하는 것도 성능 향상에 도움이 됨
- 알고리즘 튜닝을 위한 성능 최적화 방법은 하이퍼파라미터에 대한 경우의 수를 모두 고려해야 하기 때문에 모델 훈련이 수십 번에서 수백 번 필요할 수 있음
- 성능 향상은 단시간에 해결되는 것이 아니고, 수많은 시행착오를 겪어야 함



● 성능 최적화

Robot Media Laboratory

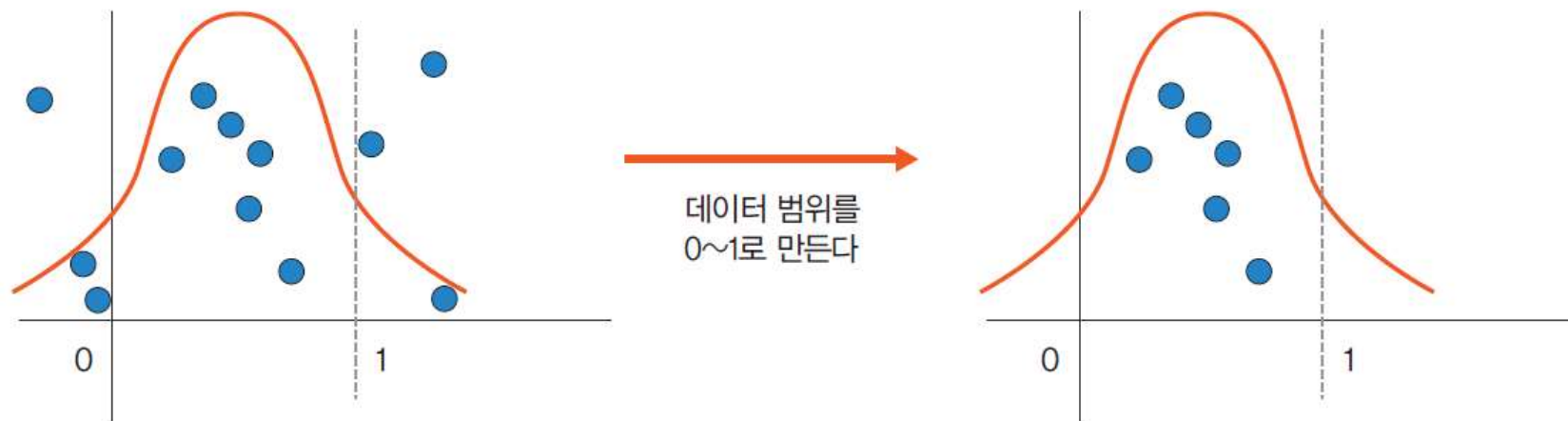
● 하이퍼파라미터를 이용한 성능 최적화

- 배치 정규화, 드롭아웃, 조기 종료 있음

● 배치 정규화를 이용한 성능 최적화

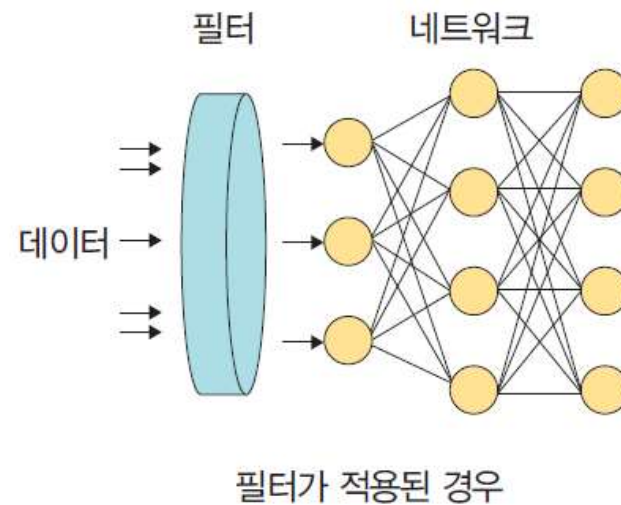
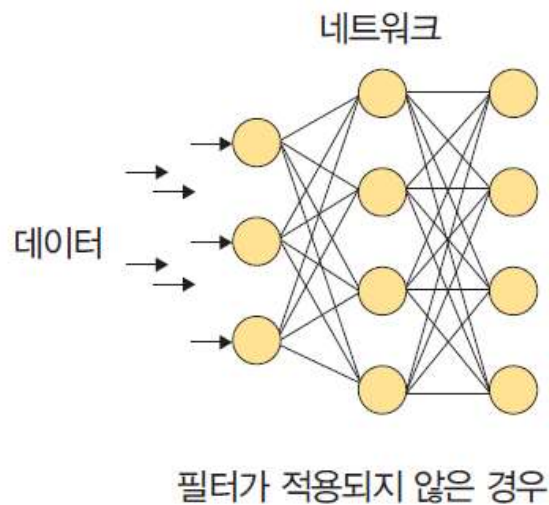
● 정규화(normalization)

- 데이터 범위를 사용자가 원하는 범위로 제한하는 것을 의미
- 각 특성 범위(스케일(scale))를 조정한다는 의미로 특성 스케일링(feature scaling)



● 규제화(regularization)

- 모델 복잡도를 줄이기 위해 제약을 두는 방법
- 제약은 데이터가 네트워크에 들어가기 전에 필터를 적용한 것
- 규제를 이용하여 모델 복잡도를 줄이는 방법
- 드롭아웃, 조기 종료

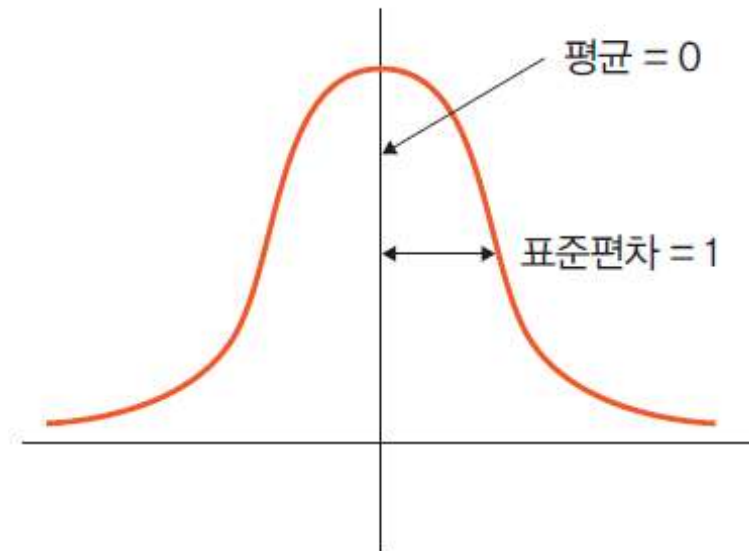


● 성능 최적화

Robot Media Laboratory

● 표준화(standardization)

- 기존 데이터를 평균은 0, 표준편차는 1인 형태의 데이터로 만드는 방법
- 표준화 스칼라(standard scaler) 혹은 z-스코어 정규화(z-score normalization)



● 성능 최적화

Robot Media Laboratory

● 배치 정규화(batch normalization)

- 데이터 분포가 안정되어 학습 속도를 높일 수 있음
- 배치 정규화는 기울기 소멸(gradient vanishing)이나 기울기 폭발(gradient exploding) 같은 문제를 해결하기 위한 방법
- 일반적으로 기울기 소멸이나 폭발 문제를 해결하기 위해 손실 함수로 렐루(ReLU)를 사용하거나 초깃값 튜닝, 학습률(learning rate) 등을 조정
- 단계마다 활성화 함수를 거치면서 데이터셋 분포가 일정해지기 때문에 속도를 향상



● 성능 최적화

Robot Media Laboratory

● 기울기 소멸

- 오차 정보를 역전파시키는 과정에서 기울기가 급격히 0에 가까워져 학습이 되지 않는 현상

● 기울기 폭발

- 학습 과정에서 기울기가 급격히 커지는 현상

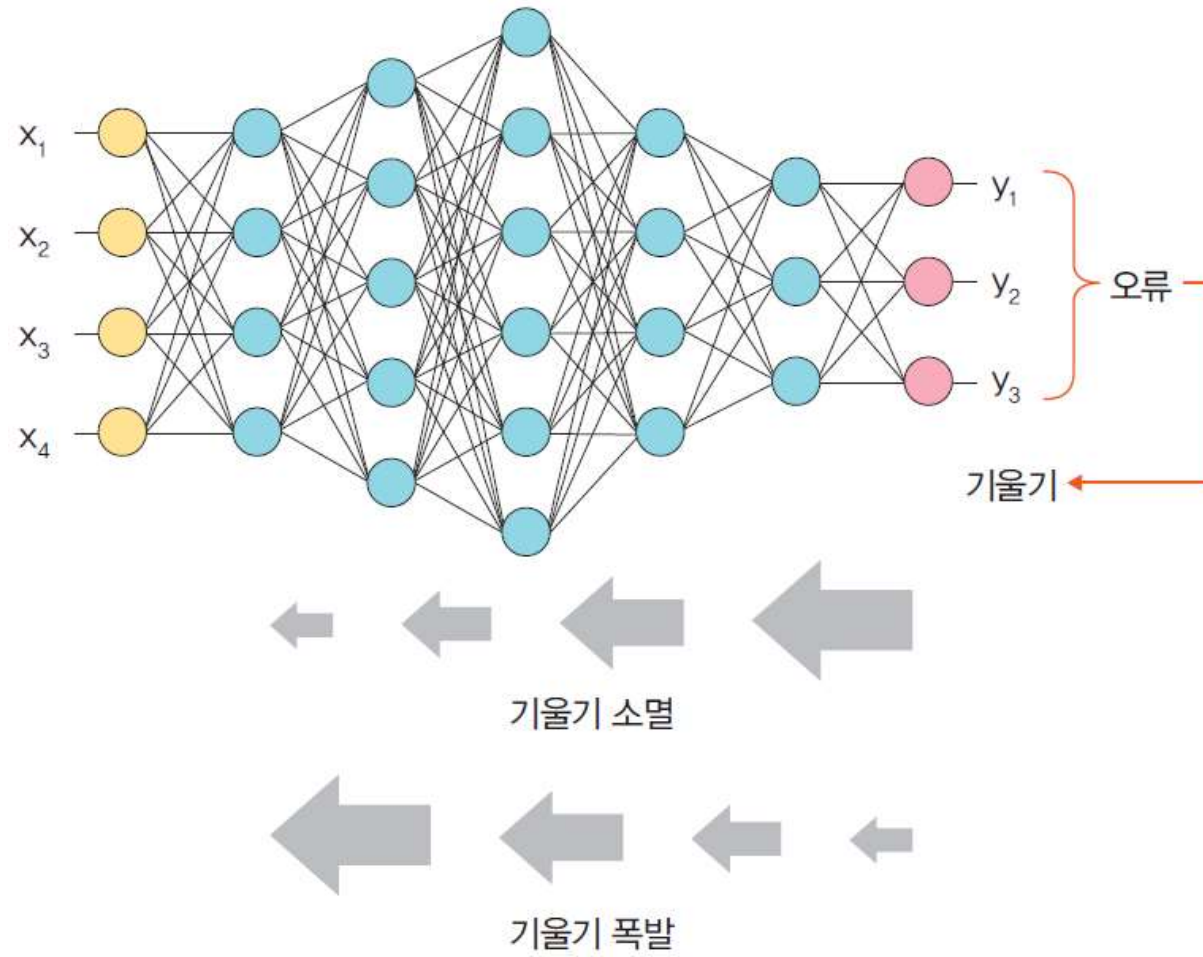
● 기울기 소멸과 폭발 원인

- 내부 공변량 변화(internal covariance shift)
- 네트워크의 각 층마다 활성화 함수가 적용되면서 입력 값들의 분포가 계속 바뀌는 현상



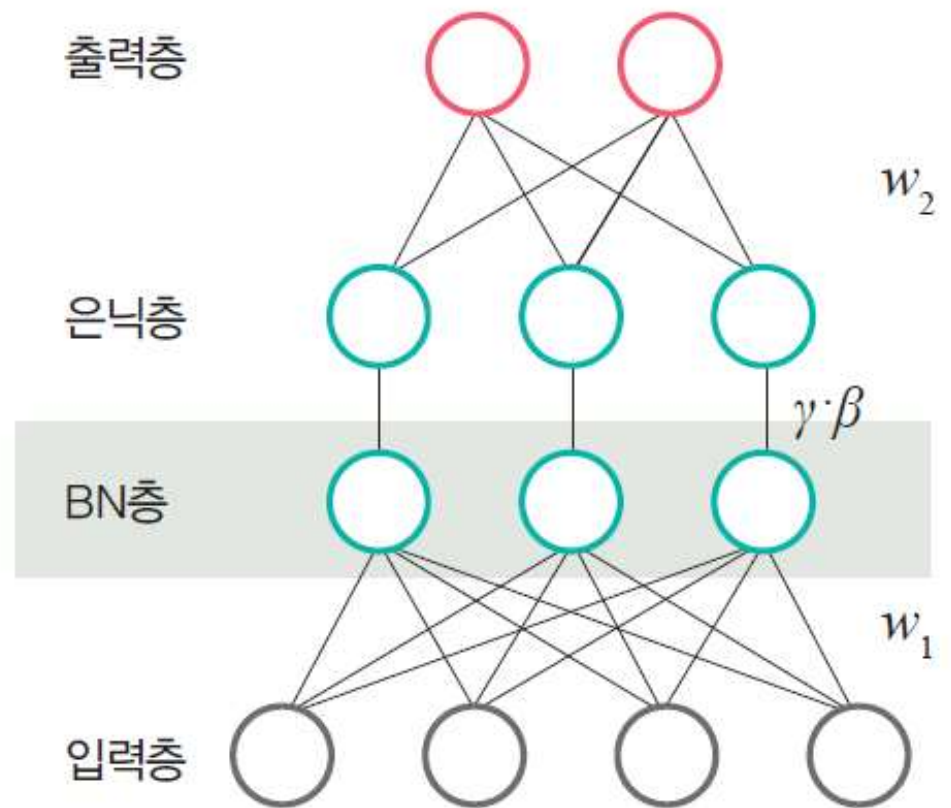
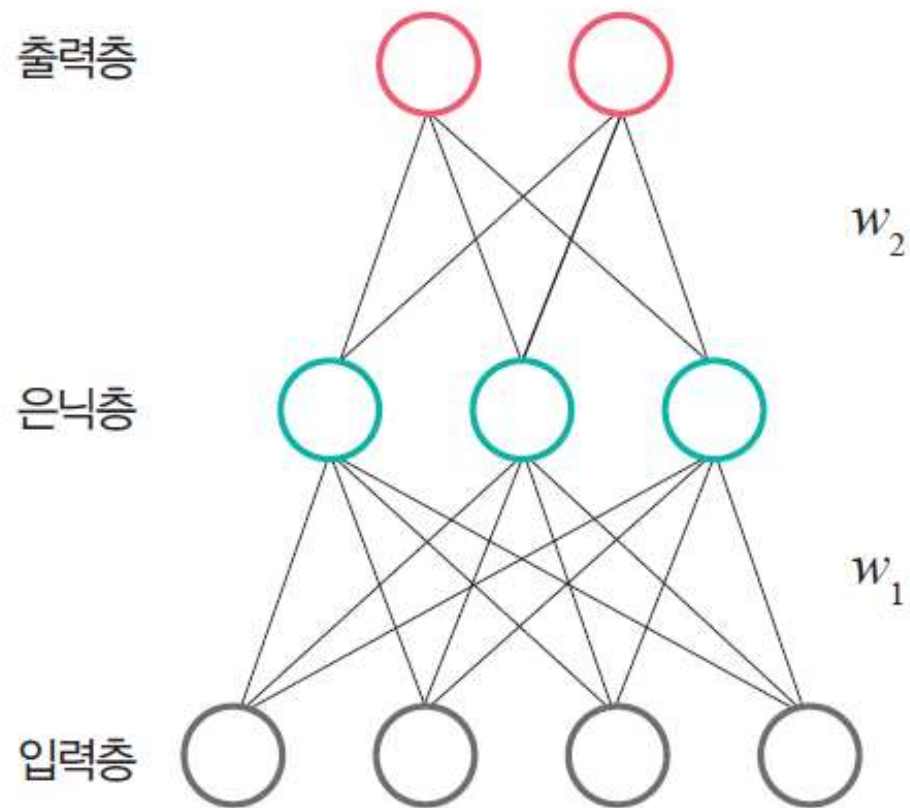
● 성능 최적화

Robot Media Laboratory



● 성능 최적화

Robot Media Laboratory



● 성능 최적화

Robot Media Laboratory

● 배치 정규화(batch normalization)

- 분산된 분포를 정규 분포로 만들기 위해 표준화와 유사한 방식을 미니 배치(mini-batch)에 적용하여 평균은 0으로, 표준편차는 1로 유지하도록함

① 미니 배치 평균을 구함

$$\mu\beta \leftarrow \frac{1}{m} \sum_{i=1}^m x_i \quad \text{---①}$$

② 미니 배치의 분산과 표준편차를 구함

$$\sigma^2\beta \leftarrow \frac{1}{m} \sum_{i=1}^m (x_i - \mu\beta)^2 \quad \text{---②}$$

③ 정규화를 수행

$$\hat{x}_i \leftarrow \frac{x_i - \mu\beta}{\sqrt{\sigma^2\beta + \epsilon}} \quad \text{---③}$$

④ 스케일(scale)을 조정(데이터 분포 조정)함

$$y_i \leftarrow \gamma \hat{x}_i + \beta \Leftrightarrow BN_{\gamma,\beta}(x_i) \quad \text{---④}$$

$$\left(\begin{array}{l} \text{입력: } \beta = \{x_1, x_2, \dots, x_n\} \\ \text{학습해야 할 하이퍼파라미터: } \gamma, \beta \\ \text{출력: } y_i = BN_{\gamma,\beta}(x_i) \end{array} \right)$$



● 성능 최적화

Robot Media Laboratory

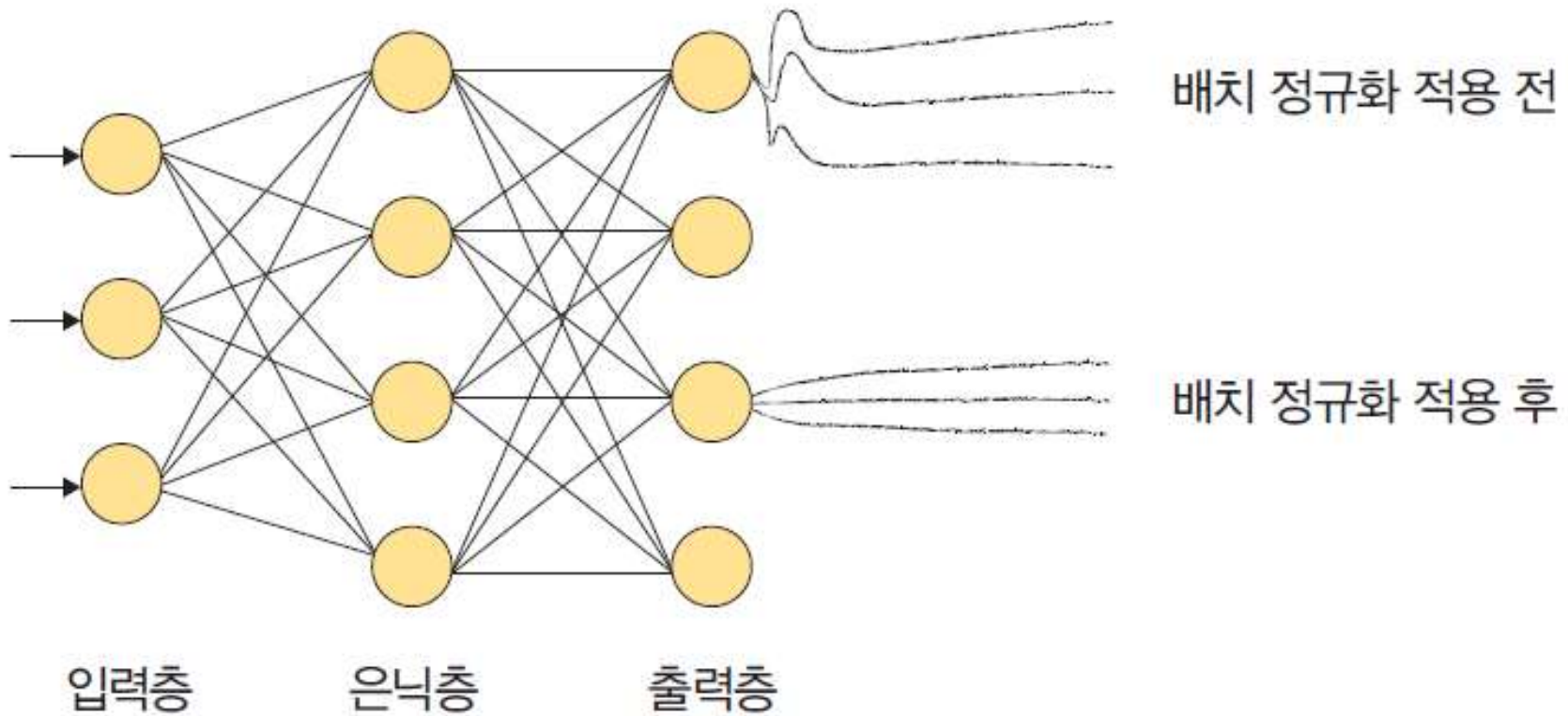
● 배치 정규화(batch normalization) 단점

- 배치 크기가 작을 때는 정규화 값이 기존 값과 다른 방향으로 훈련될 수 있음
- RNN은 네트워크 계층별로 미니 정규화를 적용해야 하기 때문에 모델이 더 복잡해지면서 비효율적
- 문제들을 해결하기 위한 가중치 수정, 네트워크 구성 변경 등을 수행
- 배치 정규화를 적용하면 적용하지 않았을 때보다 성능이 좋아지기 때문에 많이 사용
- 신경망의 층이 깊어질수록 학습할 때 가정했던 입력 분포가 변화하여 엉뚱한 학습이 진행될 수 있음
- 배치 정규화를 적용해서 입력 분포를 고르게 맞추어 주면서 과적합을 해소



● 성능 최적화

Robot Media Laboratory

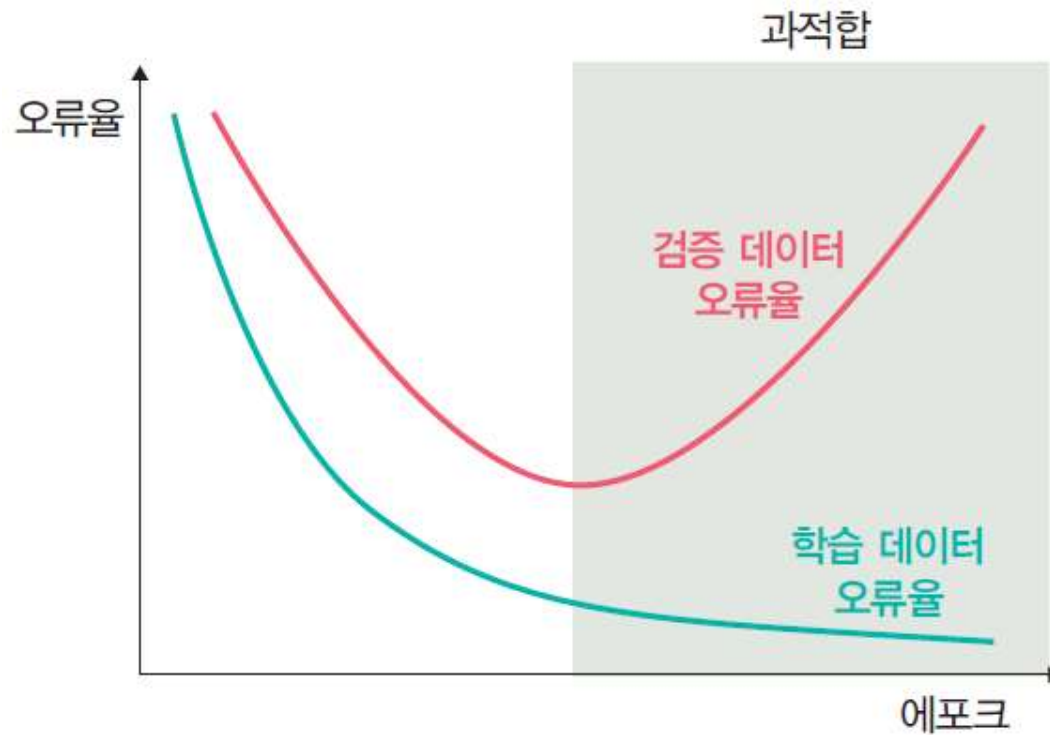


● 성능 최적화

Robot Media Laboratory

● 드롭아웃을 이용한 성능 최적화

- 훈련 데이터셋에 대해 훈련을 계속한다면 오류는 줄어들지만 검증 데이터셋에 대한 오류는 어느 순간부터 증가 -> 과적합



● 성능 최적화

Robot Media Laboratory

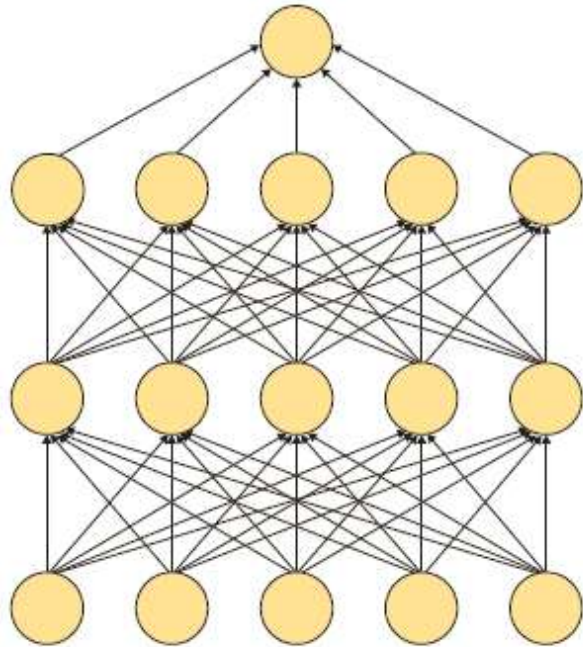
● 드롭아웃(dropout)

- 훈련할 때 일정 비율의 뉴런만 사용하고, 나머지 뉴런에 해당하는 가중치는 업데이트하지 않는 방법
- 매 단계마다 사용하지 않는 뉴런을 바꾸어 가며 훈련시킴
- 노드를 임의로 끄면서 학습하는 방법으로, 은닉층에 배치된 노드 중 일부를 임의로 끄면서 학습
- 꺼진 노드는 신호를 전달하지 않으므로 지나친 학습을 방지
- 적절히 사용되는 드롭아웃은 성능을 향상시키는 데 도움이 되므로 모델 최적화에 활용하기 좋음

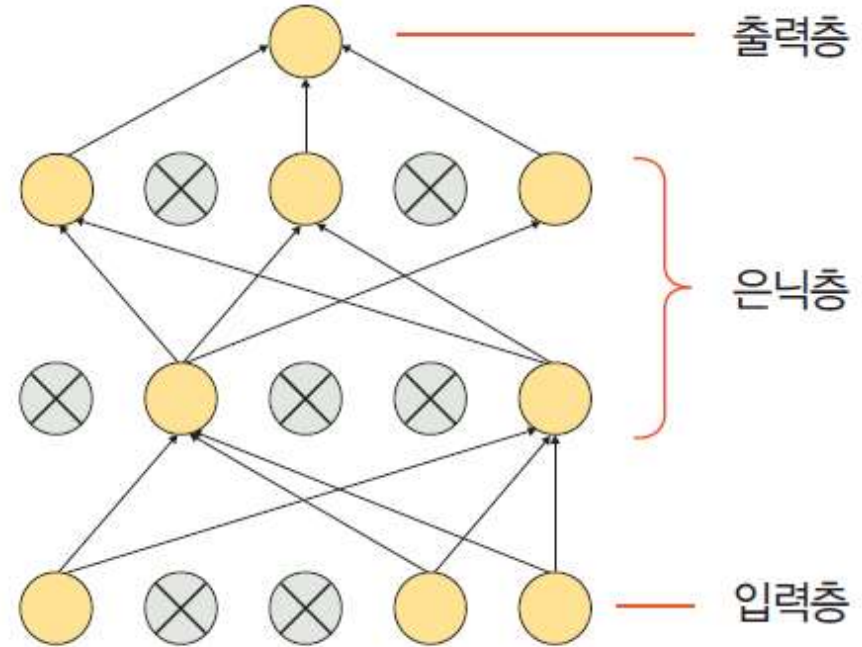


● 성능 최적화

Robot Media Laboratory



일반적인 신경망



드롭아웃이
적용된 신경망



● 성능 최적화

Robot Media Laboratory

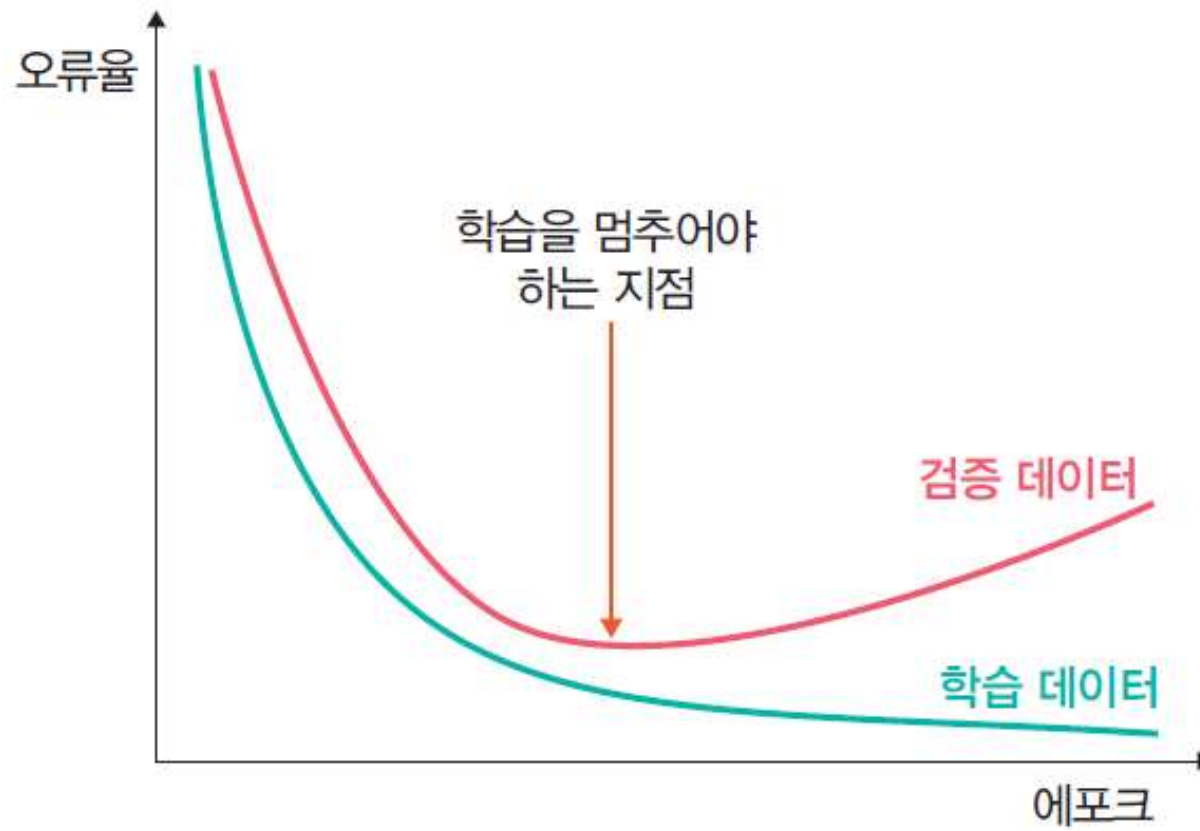
● 조기 종료(early stopping)

- 뉴럴 네트워크가 과적합을 회피하는 규제 기법
- 훈련 데이터와 별도로 검증 데이터를 준비하고, 매 에포크마다 검증 데이터에 대한 손실 (validation loss)을 측정하여 모델의 종료 시점을 제어함
- 조기 종료는 검증에 대한 손실이 증가하는 시점에서 훈련을 멈추도록 조정함



● 성능 최적화

Robot Media Laboratory



순환신경망

RNN



● RNN

Robot Media Laboratory

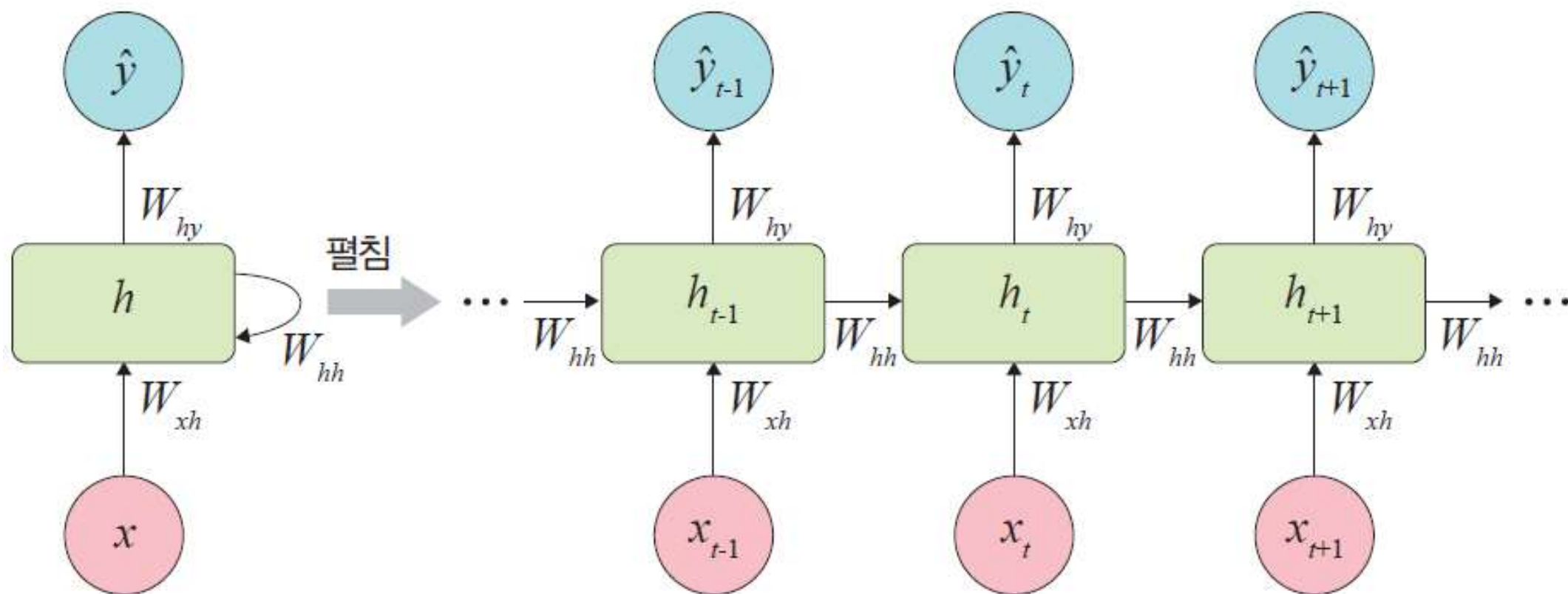
● RNN(Recurrent Neural Network)

- 시간적으로 연속성이 있는 데이터를 처리하려고 고안된 인공 신경망
- 은닉층 노드들이 연결되어 이전 단계 정보를 은닉층 노드에 저장할 수 있도록 구성한 신경망
- 'Recurrent(반복되는)'는 이전 은닉층이 현재 은닉층의 입력이 되면서 '반복되는 순환 구조를 갖는다'는 의미
- RNN이 기존 네트워크와 다른 점은 '기억(memory)'을 갖는다는 것
- 기억은 현재까지 입력 데이터를 요약한 정보
- 새로운 입력이 네트워크로 들어올 때마다 기억은 조금씩 수정되며, 결국 최종적으로 남겨진 기억은 모든 입력 전체를 요약한 정보



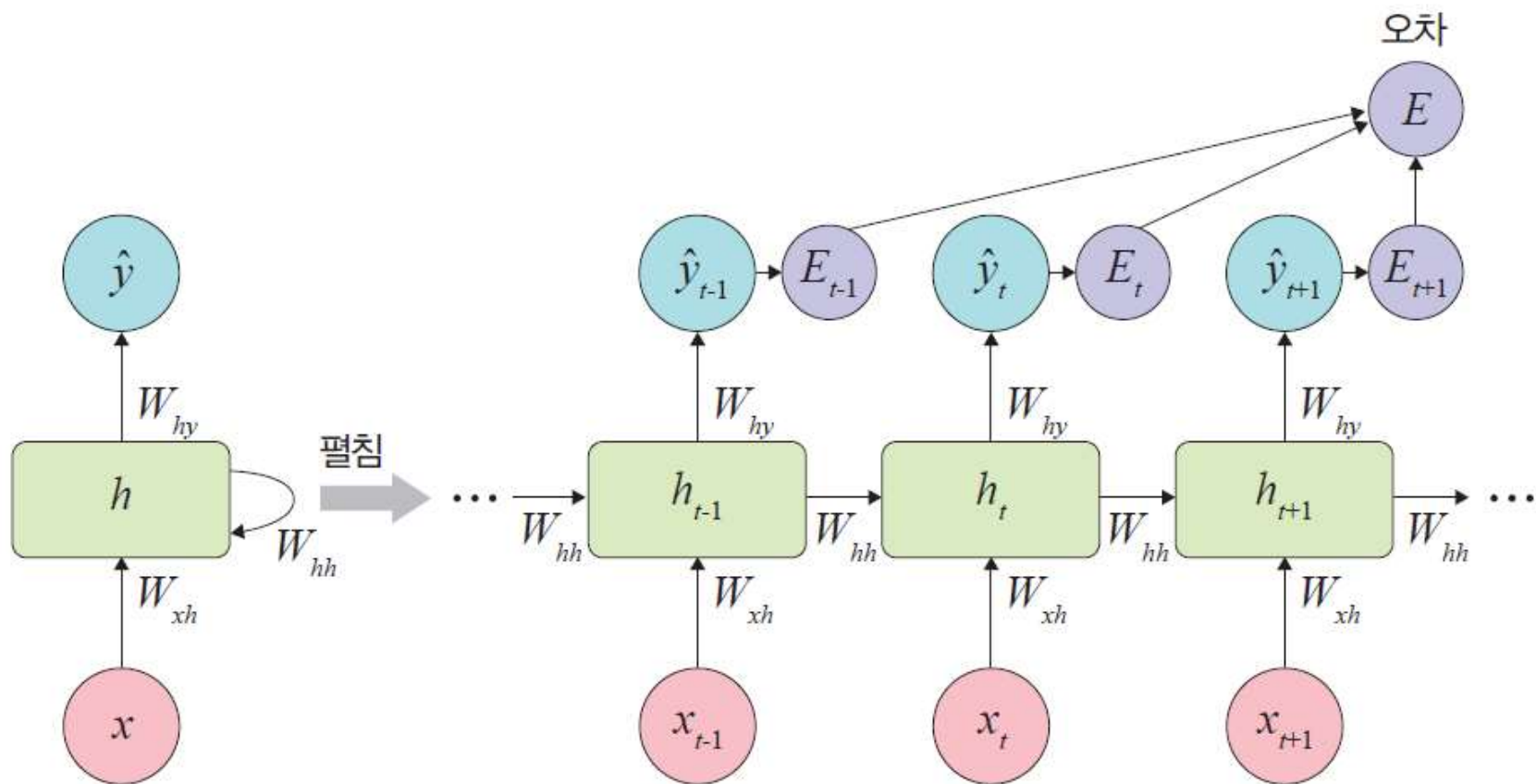
RNN

Robot Media Laboratory



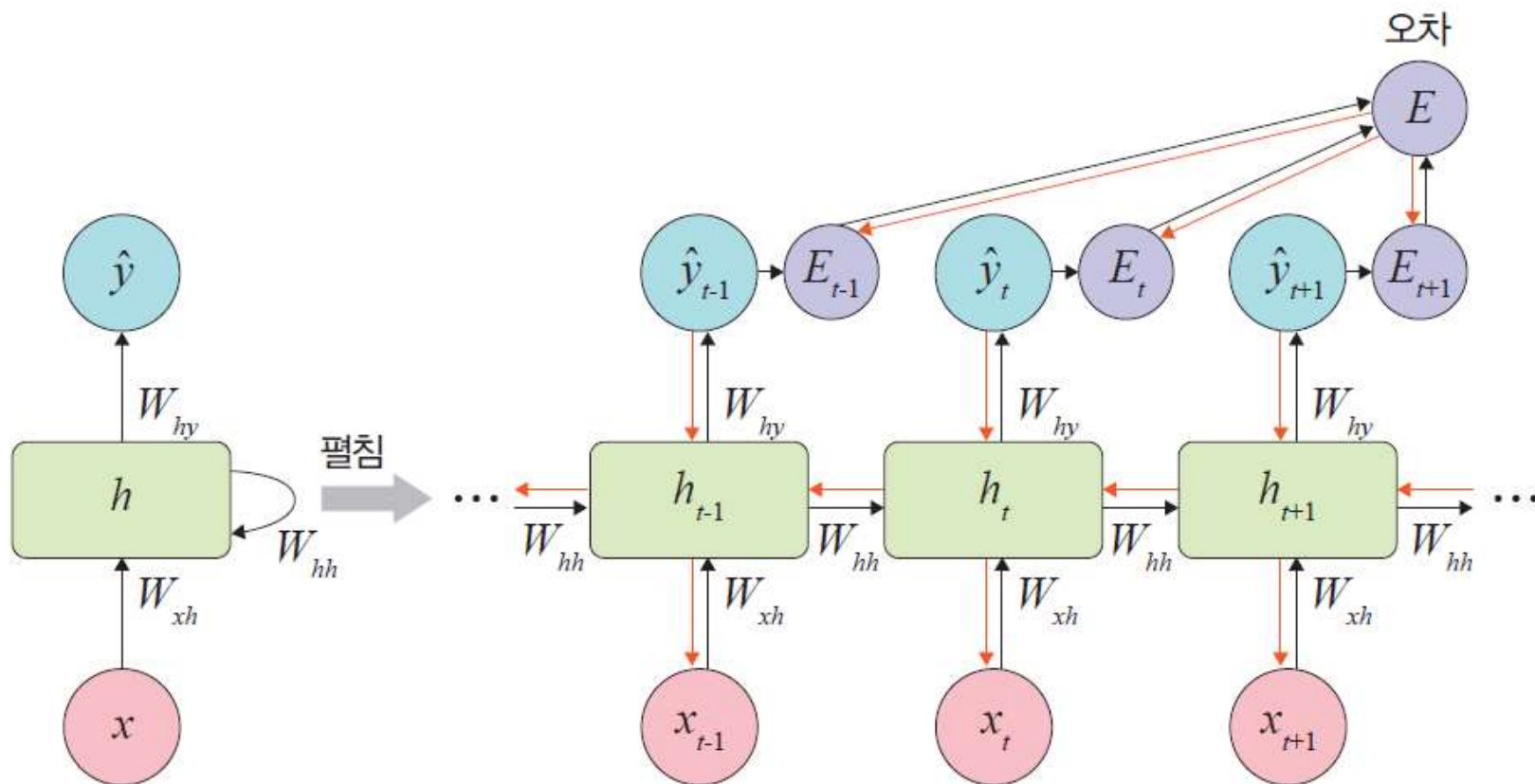
RNN

Robot Media Laboratory



RNN

Robot Media Laboratory



● RNN

Robot Media Laboratory

● 일대일

- 순환이 없기 때문에 RNN이라고 말하기 어려우며, 순방향 네트워크가 대표적 사례

● 일대다

- 입력이 하나이고, 출력이 다수인 구조
- 이미지를 입력해서 이미지에 대한 설명을 문장으로 출력하는 이미지 캡션(image captioning)

● 다대일

- 입력이 다수이고 출력이 하나인 구조로, 문장을 입력해서 긍정/부정을 출력하는 감성분석기

● 다대다

- 입력과 출력이 다수인 구조로, 언어를 번역하는 자동 번역기 등

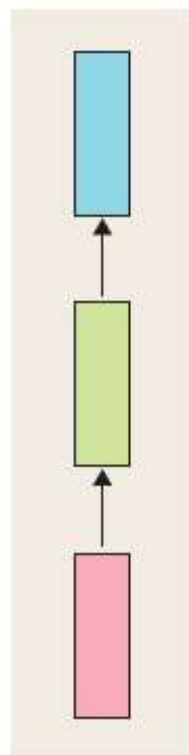
● 동기화 다대다

- 문장에서 다음에 나올 단어를 예측하는 언어 모델, 즉 프레임 수준의 비디오 분류

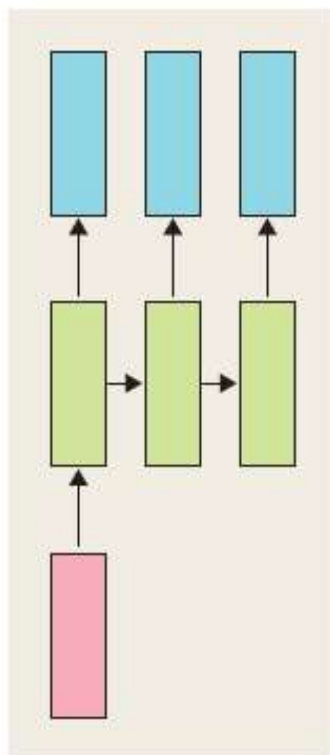


RNN

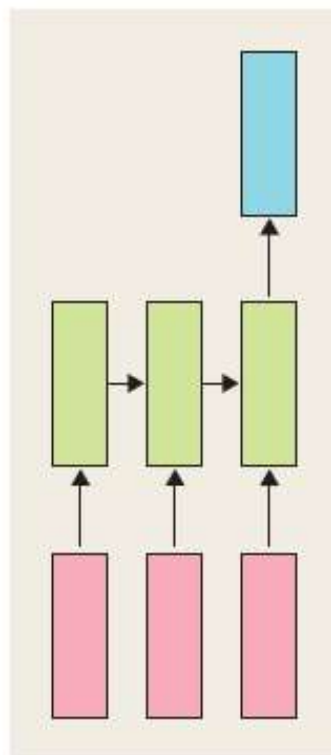
Robot Media Laboratory



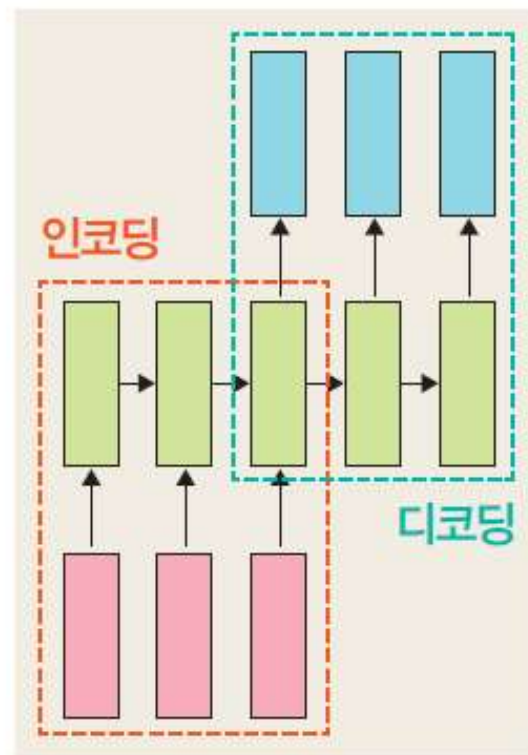
일대일



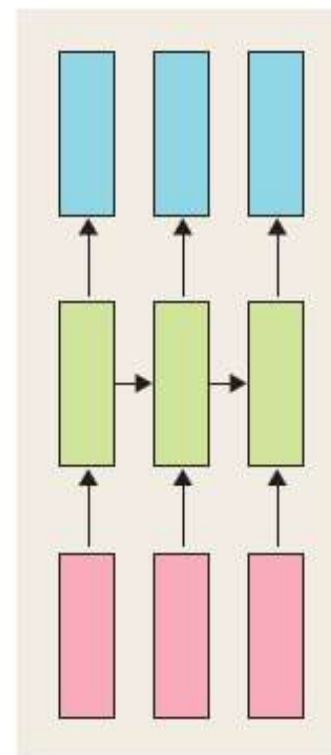
일대다



다대일



다대다



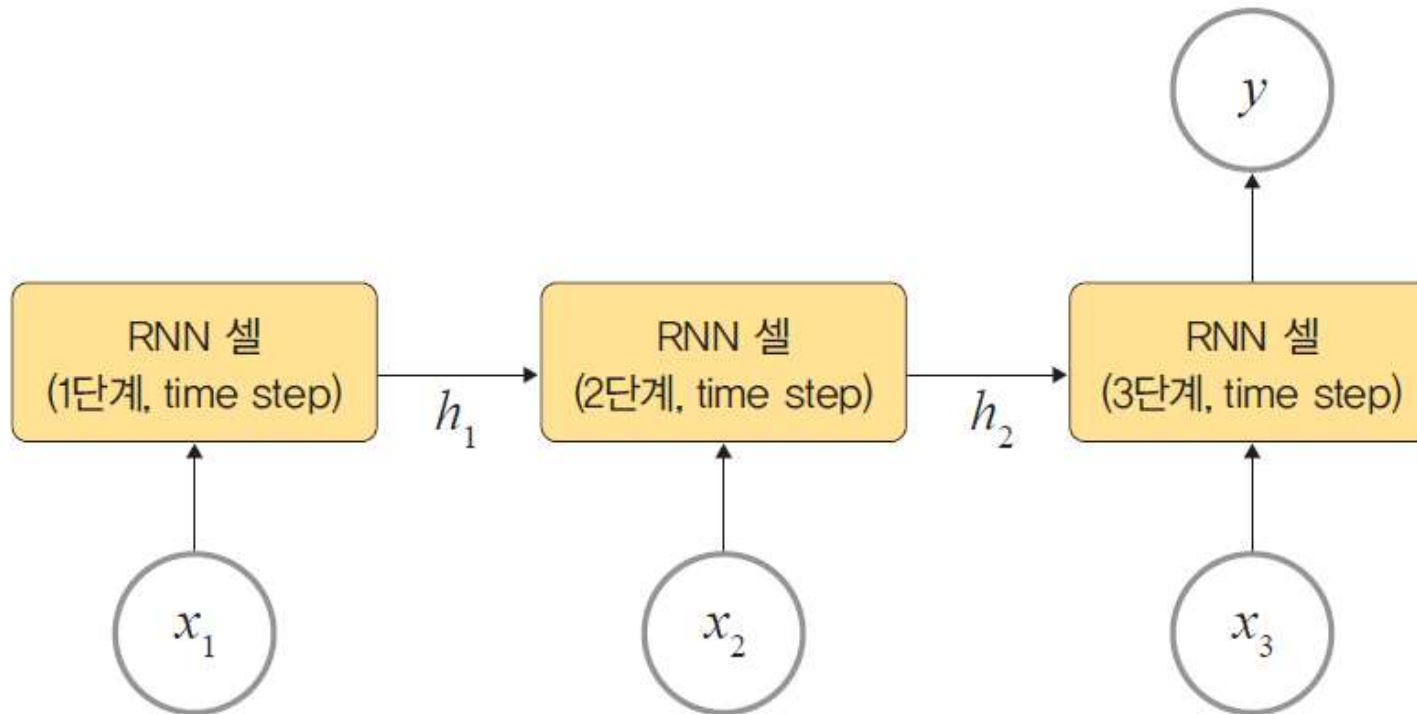
동기화 다대다



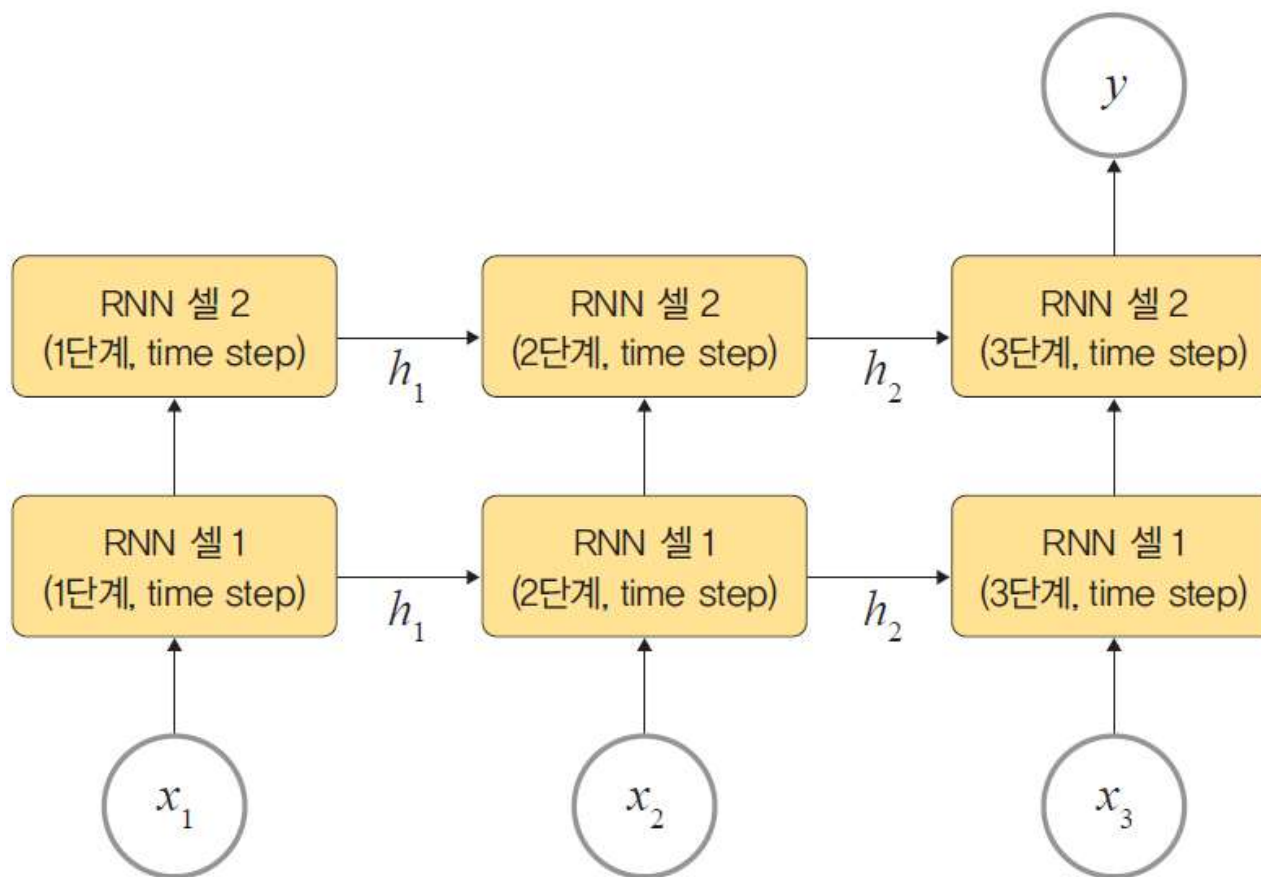
● RNN

Robot Media Laboratory

● 다대일



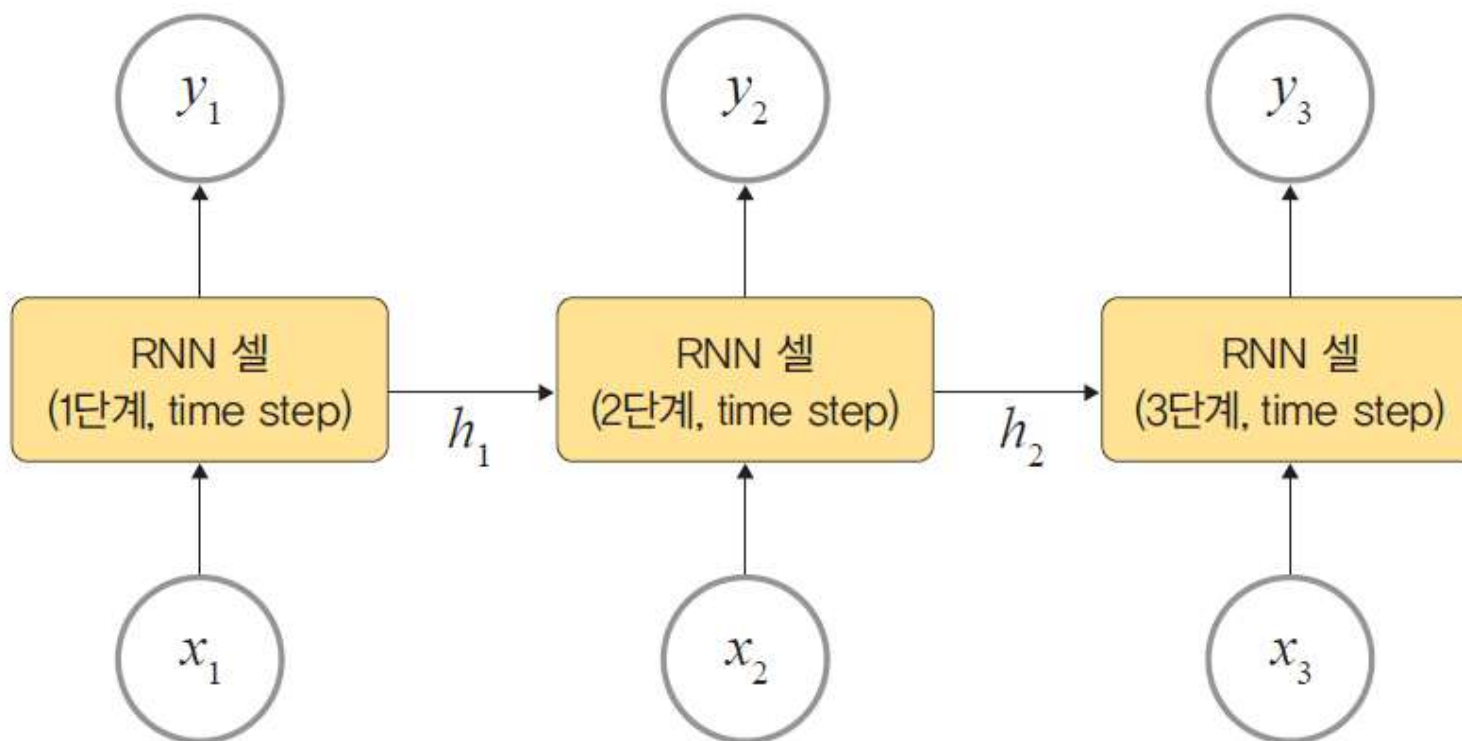
● 다층 다대일



● RNN

Robot Media Laboratory

● 다대다



● RNN

Robot Media Laboratory

● LSTM

- RNN은 결정적 단점이 있음
- 망각 게이트(forget gate)는 과거 정보를 어느 정도 기억할지 결정
- 과거 정보와 현재 데이터를 입력받아 시그모이드를 취한 후 그 값을 과거 정보에 곱해 줌
- 시그모이드의 출력이 0이면 과거 정보는 버리고, 1이면 과거 정보는 온전히 보존

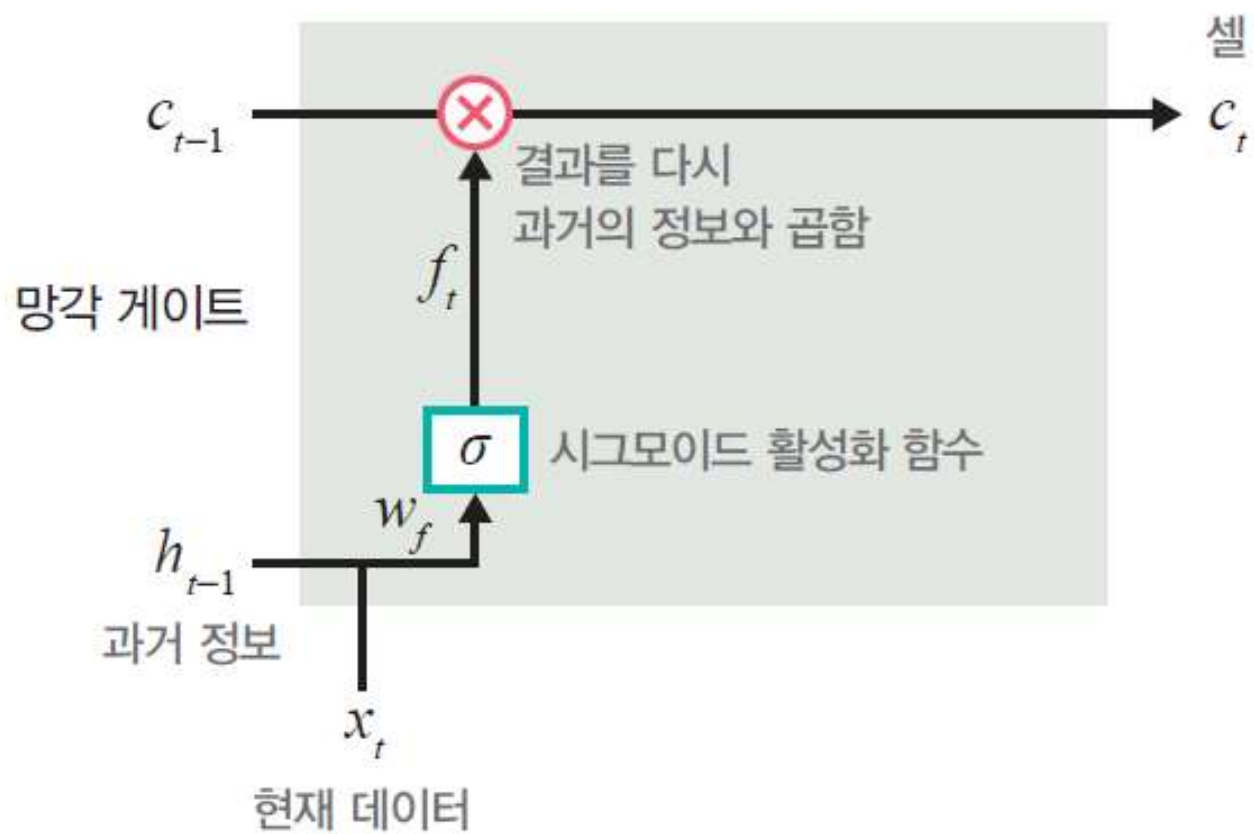
$$f_t = \sigma(w_f[h_{t-1}, x_t])$$

$$c_t = f_t \cdot c_{t-1}$$



RNN

Robot Media Laboratory



● RNN

Robot Media Laboratory

● 입력 게이트(input gate)

- 현재 정보를 기억하기 위해 만들어졌음
- 과거 정보와 현재 데이터를 입력받아 시그모이드와 하이퍼볼릭 탄젠트 함수를 기반으로 현재 정보에 대한 보존량을 결정
- 즉, 현재 메모리에 새로운 정보를 반영할지 결정하는 역할을 함
- 계산한 값이 1이면 입력 x_t 가 들어올 수 있도록 허용(open)
- 계산한 값이 0이면 차단

$$i_t = \sigma(w_i[h_{t-1}, x_t])$$

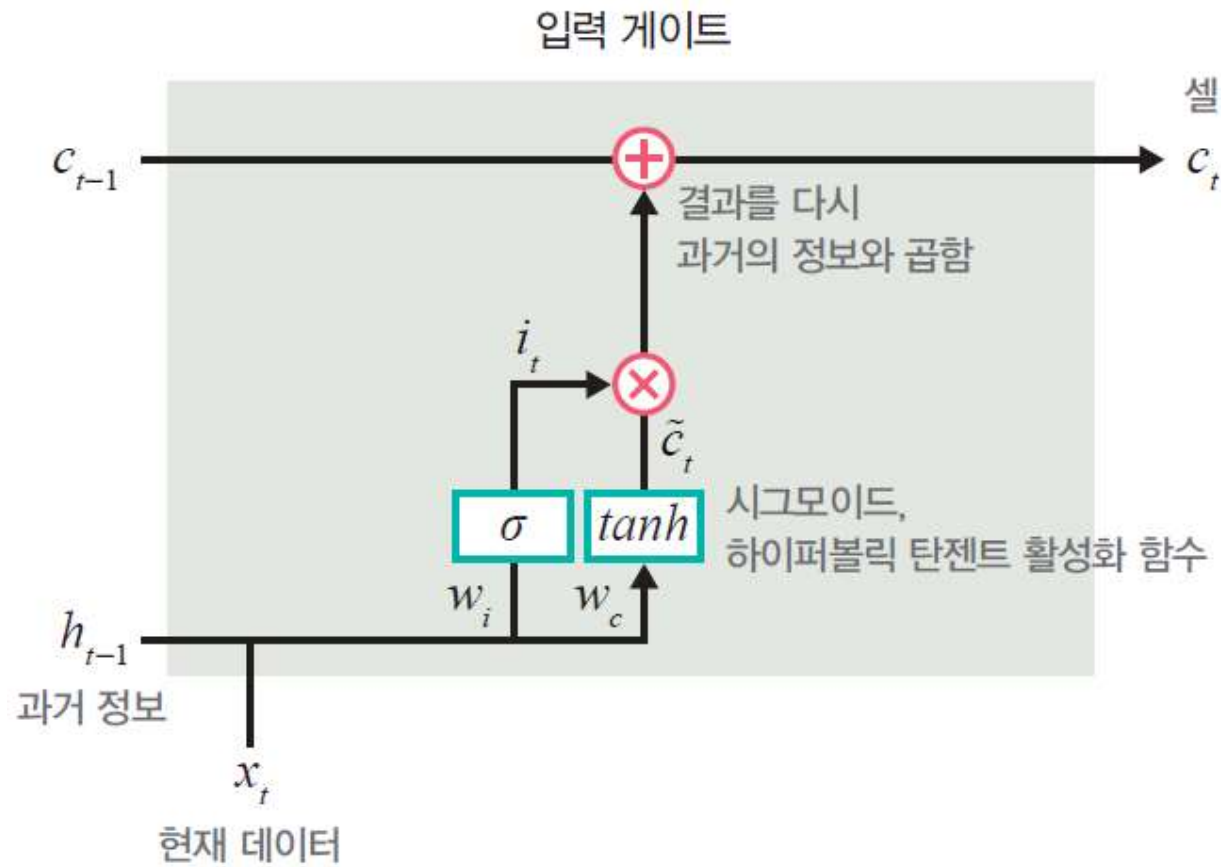
$$\tilde{c}_t = \tanh(w_c[h_{t-1}, x_t])$$

$$c_t = c_{t-1} + i_t \cdot \tilde{c}_t$$



RNN

Robot Media Laboratory



● RNN

Robot Media Laboratory

● 셀

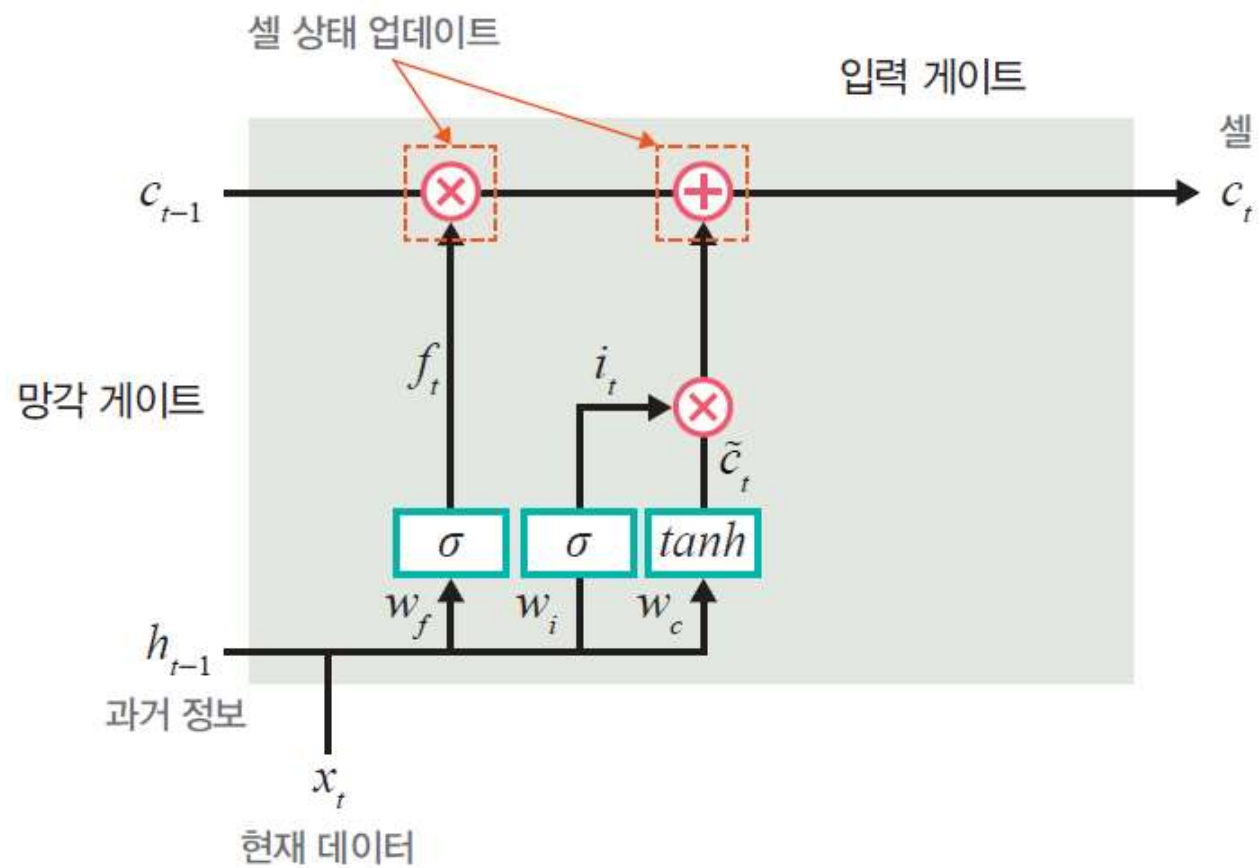
- 각 단계에 대한 은닉 노드(hidden node)를 메모리 셀이라고 함
- '총합(sum)'을 사용하여 셀 값을 반영하며, 이것으로 기울기 소멸 문제가 해결
- 망각 게이트와 입력 게이트의 이전 단계 셀 정보를 계산하여 현재 단계의 셀 상태(cell state)를 업데이트

$$i_t = \sigma(w_i[h_{t-1}, x_t])$$

$$\tilde{c}_t = \tanh(w_c[h_{t-1}, x_t])$$

$$c_t = c_{t-1} + i_t \cdot \tilde{c}_t$$





● RNN

Robot Media Laboratory

● 출력 게이트(output gate)

- 과거 정보와 현재 데이터를 사용하여 뉴런의 출력을 결정
- 이전 은닉 상태(hidden state)와 t번째 입력을 고려해서 다음 은닉 상태를 계산
- LSTM에서는 이 은닉 상태가 그 시점에서의 출력이 됨
- 출력 게이트는 갱신된 메모리의 출력 값을 제어하는 역할을 함
- 계산한 값이 1이면 의미 있는 결과로 최종 출력
- 계산한 값이 0이면 해당 연산 출력을 하지 않음

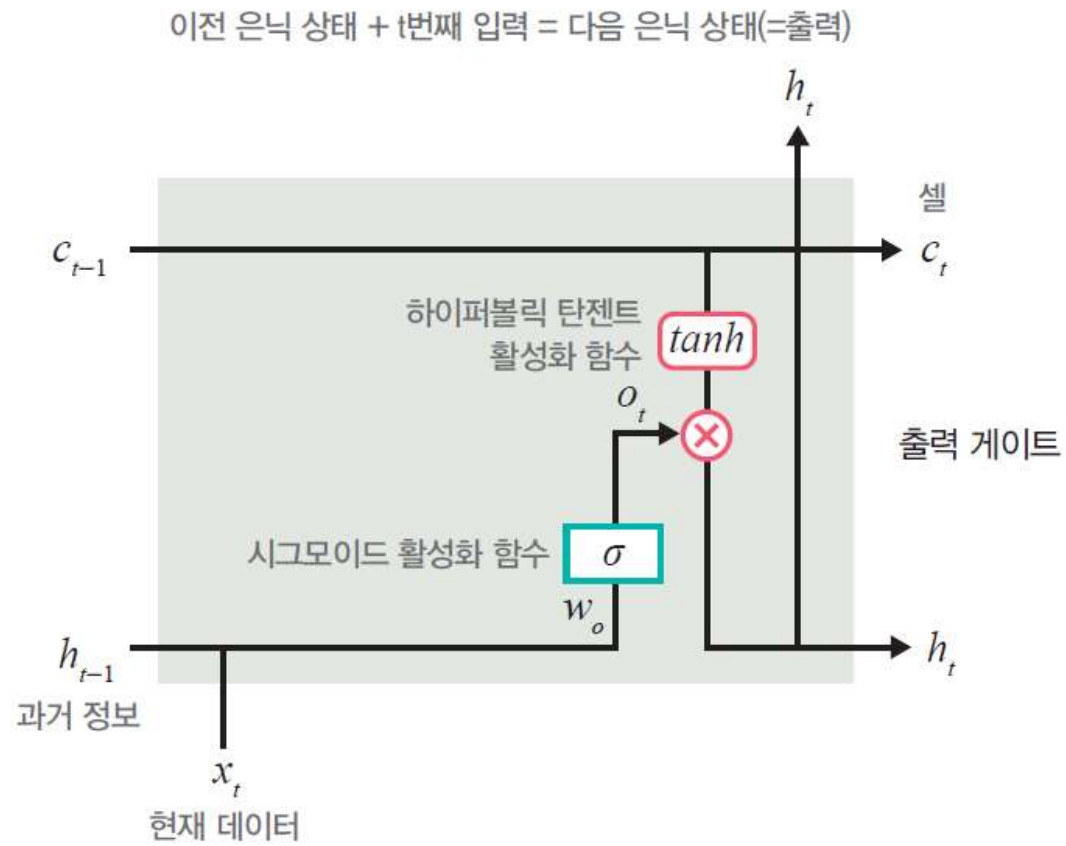
$$o_t = \sigma(w_o [h_{t-1}, x_t])$$

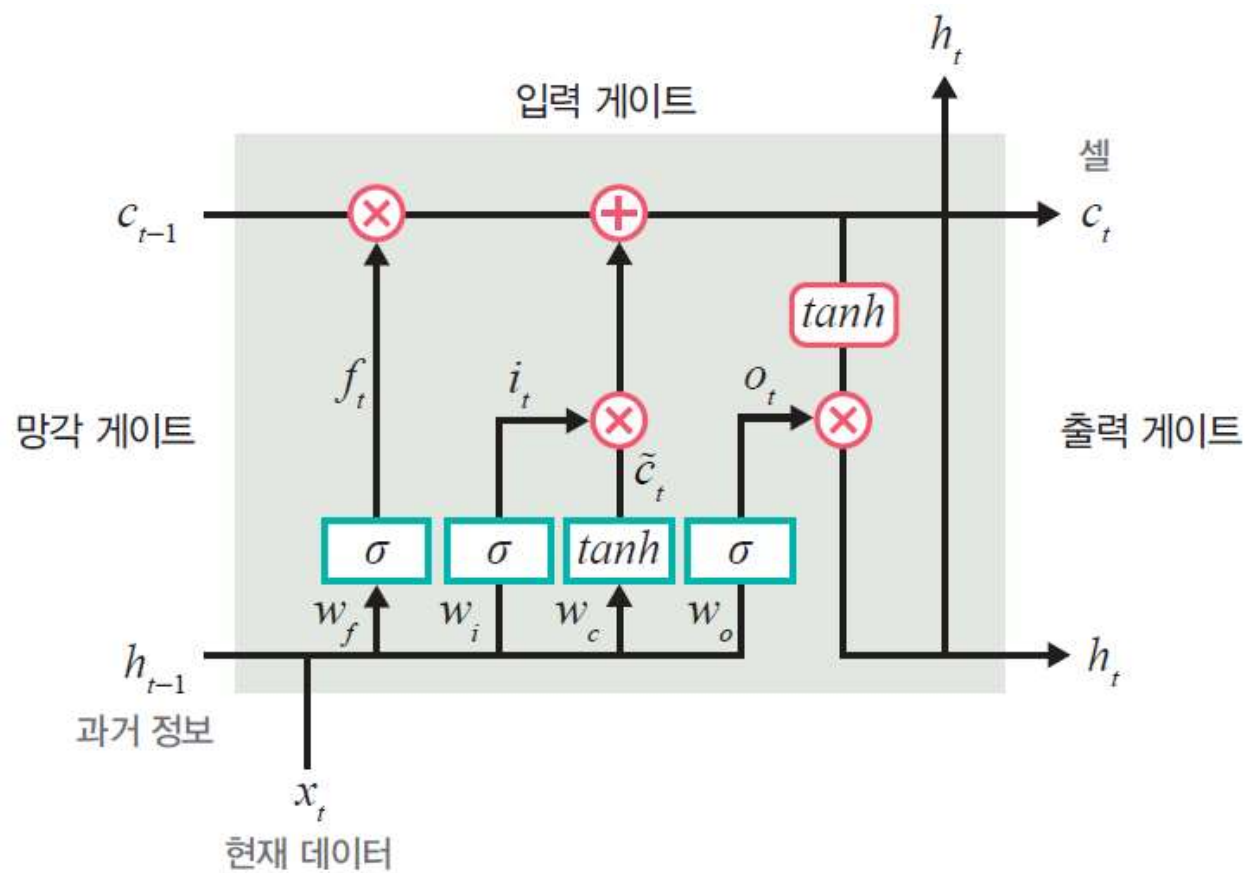
$$h_t = o_t \cdot \tanh(c_{t-1})$$



RNN

Robot Media Laboratory





● RNN

Robot Media Laboratory

● GRU(Gated Recurrent Unit)

- 게이트 메커니즘이 적용된 RNN 프레임워크의 한 종류이면서 LSTM보다 구조가 간단함
- GRU는 LSTM에서 사용하는 망각 게이트와 입력 게이트를 하나로 합친 것이며, 별도의 업데이트 게이트로 구성
- 하나의 게이트 컨트롤러(gate controller)가 망각 게이트와 입력 게이트를 모두 제어함
- 게이트 컨트롤러가 1을 출력하면 망각 게이트는 열리고 입력 게이트는 닫히며, 반대로 0을 출력하면 망각 게이트는 닫히고 입력 게이트는 열림
- 즉, 이전 기억이 저장될 때마다 단계별 입력은 삭제
- GRU는 출력 게이트가 없어 전체 상태 벡터가 매 단계마다 출력되며, 이전 상태의 어느 부분이 출력될지 제어하는 새로운 게이트 컨트롤러가 별도로 존재



● RNN

Robot Media Laboratory

● 망각 게이트(reset gate)

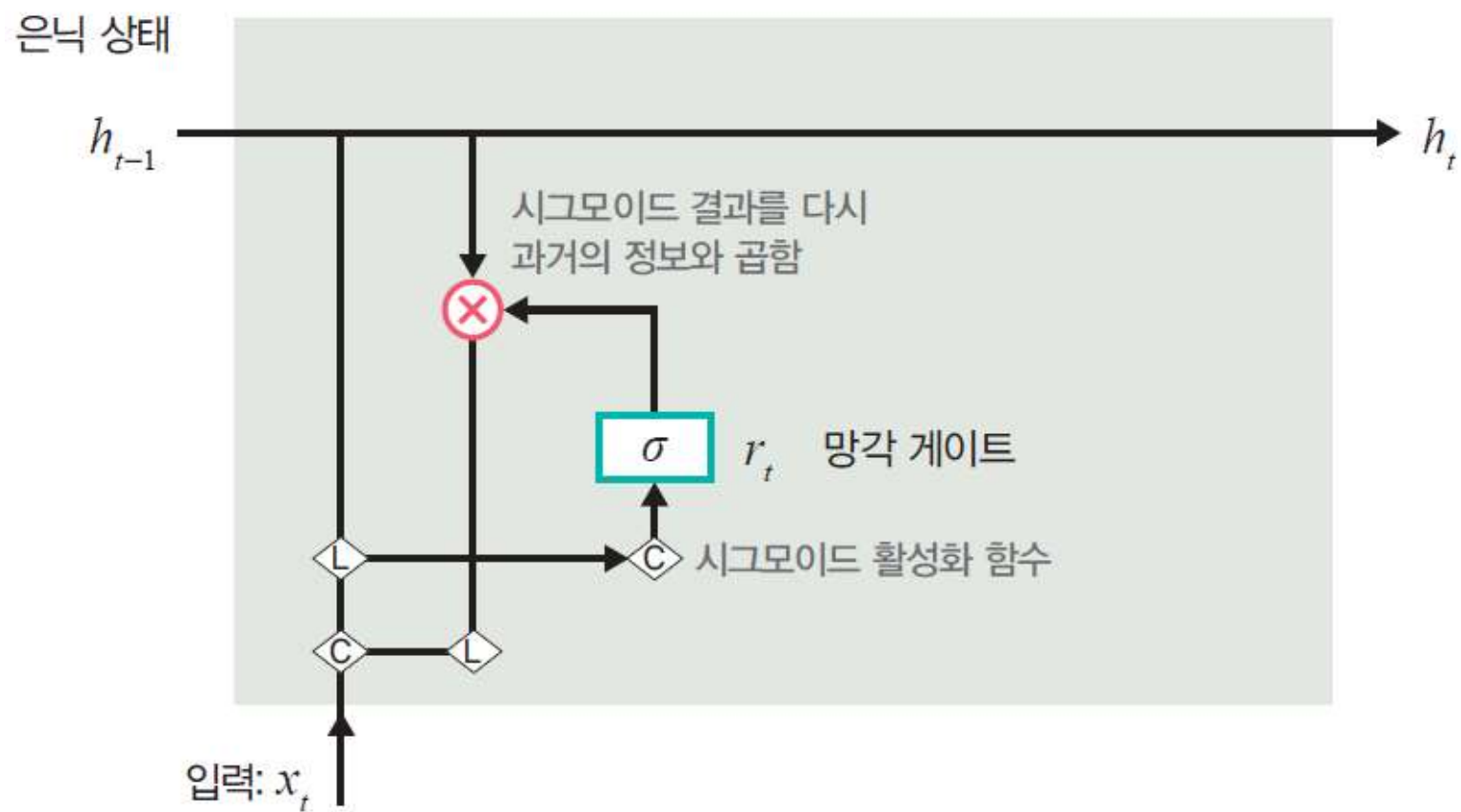
- 과거 정보를 적당히 초기화(reset)시키려는 목적으로 시그모이드 함수를 출력으로 이용하여 (0,1) 값을 이전 은닉층에 곱함

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t])$$



RNN

Robot Media Laboratory



● RNN

Robot Media Laboratory

● 업데이트 게이트(update gate)

- 과거와 현재 정보의 최신화 비율을 결정하는 역할을 함
- 시그모이드로 출력된 결과(z_t)는 현시점의 정보량을 결정하고 1에서 뺀 값($1 - z_t$)은 직전 시점의 은닉층 정보와 곱함

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t])$$



RNN

Robot Media Laboratory

은닉 상태

h_{t-1}

현시점의 정보량을 결정

업데이트 게이트

σ

z_t

1-

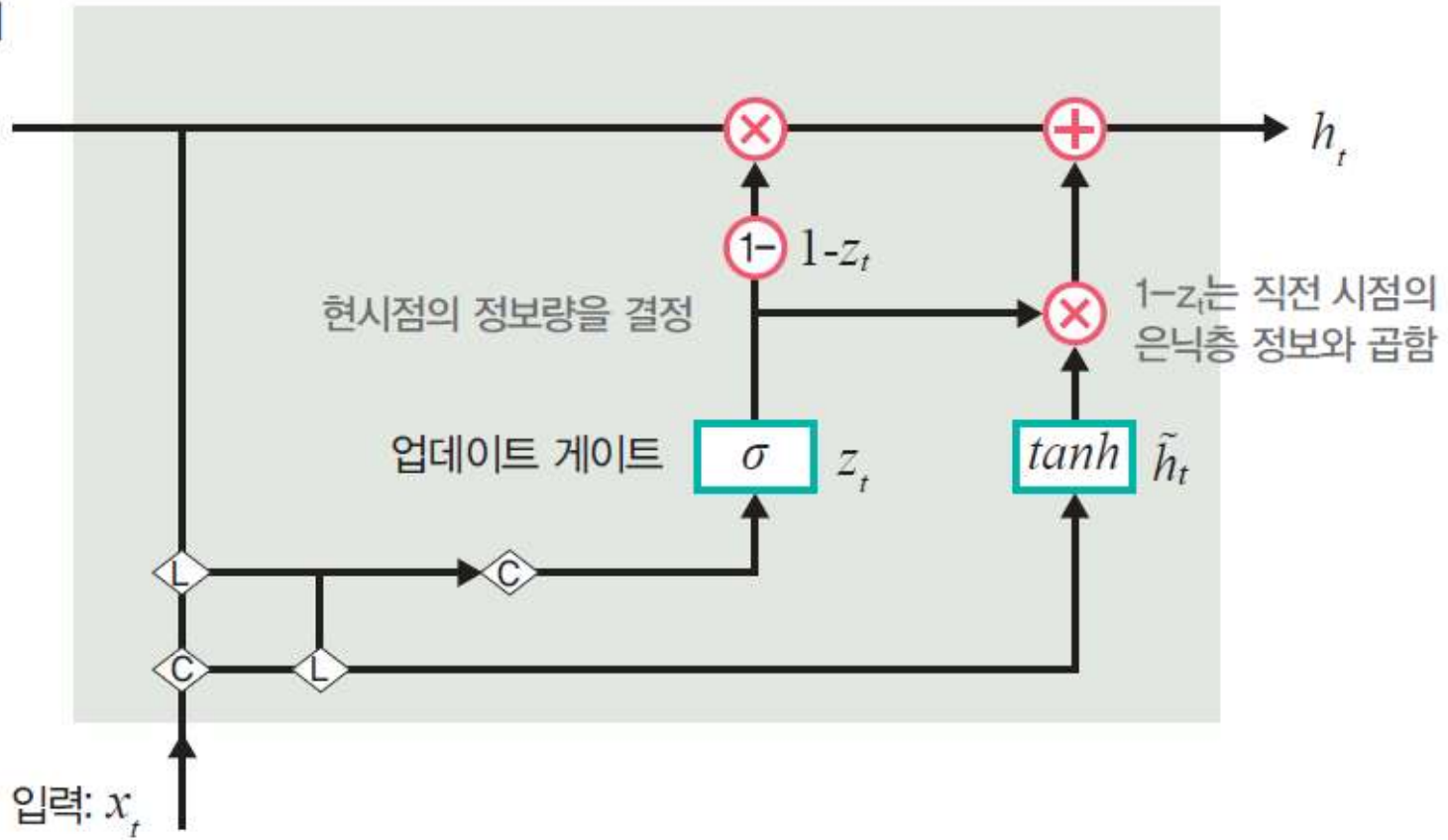
$1-z_t$

\tanh

\tilde{h}_t

$1-z_t$ 는 직전 시점의 은닉층 정보와 곱함

입력: x_t



● RNN

Robot Media Laboratory

● 후보군(candidate)

- 현시점의 정보에 대한 후보군을 계산
- 과거 은닉층의 정보를 그대로 이용하지 않고 망각 게이트의 결과를 이용하여 후보군을 계산

$$\tilde{h}_t = \tanh(W \cdot [r_t * h_{t-1}, x_t])$$

(*는 점 단위 연산(pointwise operation)입니다. 예를 들어 벡터를 더할 때 각각의 차원(dimension)에 맞게 곱하거나 더하는 것이 가능해집니다.)



● RNN

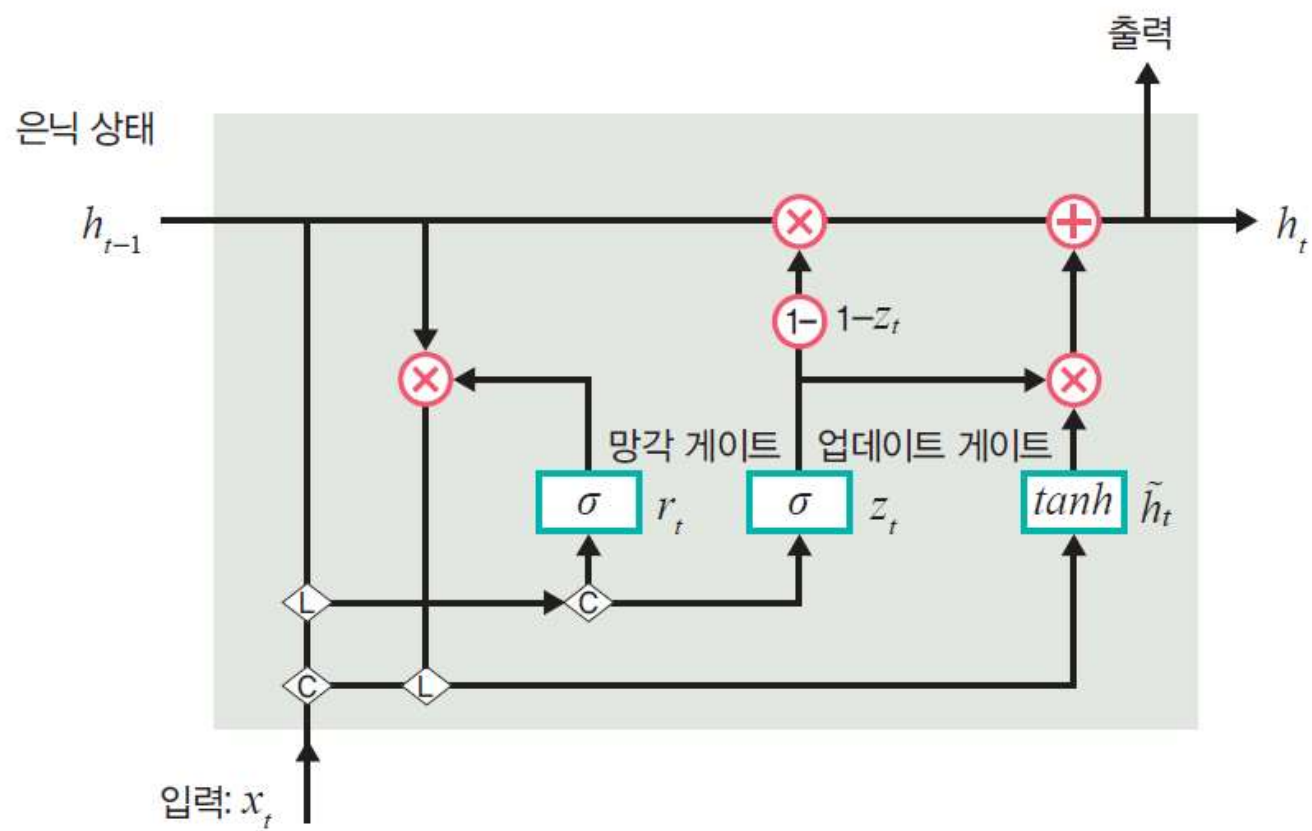
Robot Media Laboratory

● 은닉층 계산

- 마지막으로 업데이트 게이트 결과와 후보군 결과를 결합하여 현시점의 은닉층을 계산
- 시그모이드 함수의 결과는 현시점에서 결과에 대한 정보량을 결정하고, 1-시그모이드 함수의 결과는 과거의 정보량을 결정

$$h_t = (1 - z_t) * h_{t-1} + z_t \times \tilde{h}_t$$





● RNN

Robot Media Laboratory

● 양방향 RNN

- RNN은 이전 시점의 데이터들을 참고해서 정답을 예측하지만 실제 문제에서는 과거 시점이 아닌 미래 시점의 데이터에 힌트가 있는 경우도 많음
- 이전 시점의 데이터뿐만 아니라, 이후 시점의 데이터도 함께 활용하여 출력 값을 예측하고자 하는 것이 양방향 RNN(bidirectional RNN)



● RNN

Robot Media Laboratory

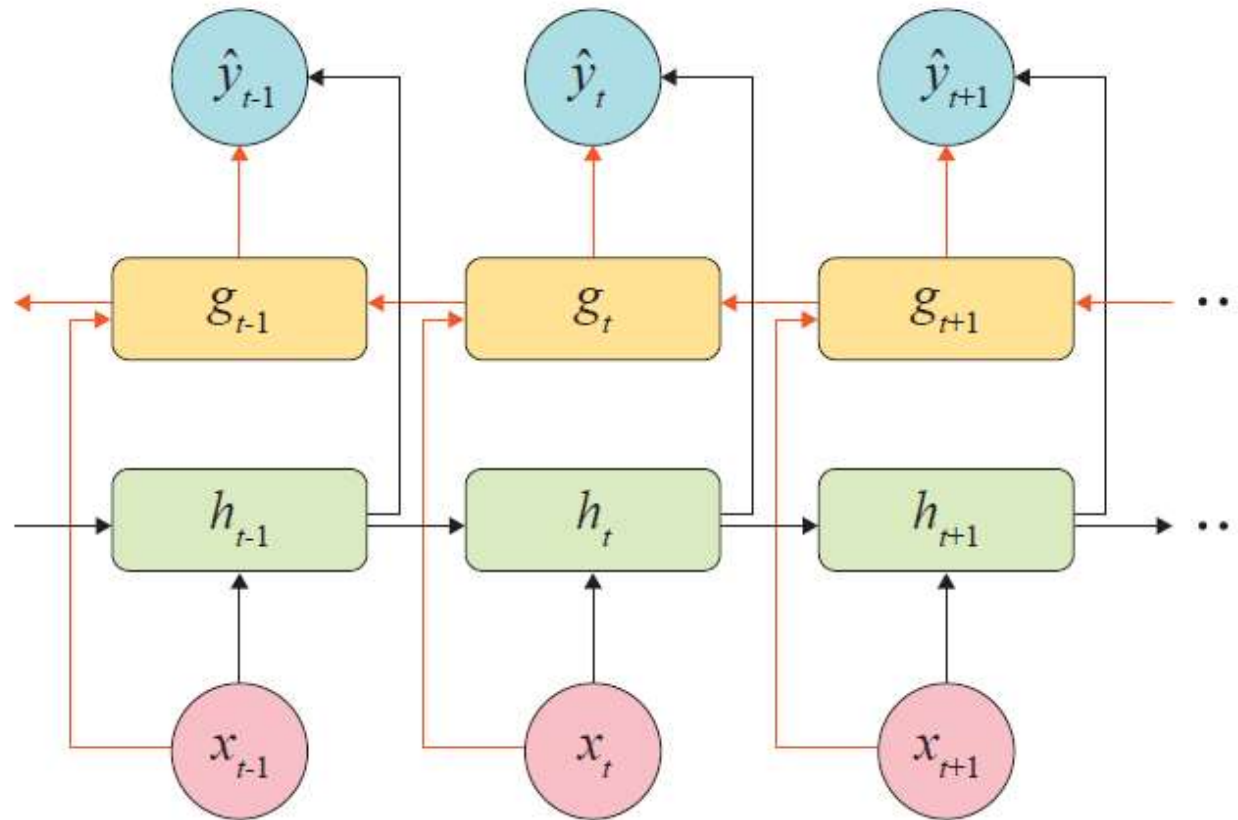
● 양방향 RNN 구조

- 양방향 RNN은 하나의 출력 값을 예측하는 데 메모리 셀 두 개를 사용
- 첫 번째 메모리 셀은 이전 시점의 은닉 상태(forward states)를 전달받아 현재의 은닉 상태를 계산
- 다음 그림에서는 초록색 메모리 셀에 해당
- 두 번째 메모리 셀은 다음 시점의 은닉 상태(backward states)를 전달받아 현재의 은닉 상태를 계산
- 다음 그림의 주황색 메모리 셀에 해당
- 이 값 두 개를 모두 출력층에서 출력 값을 예측하는 데 사용



RNN

Robot Media Laboratory



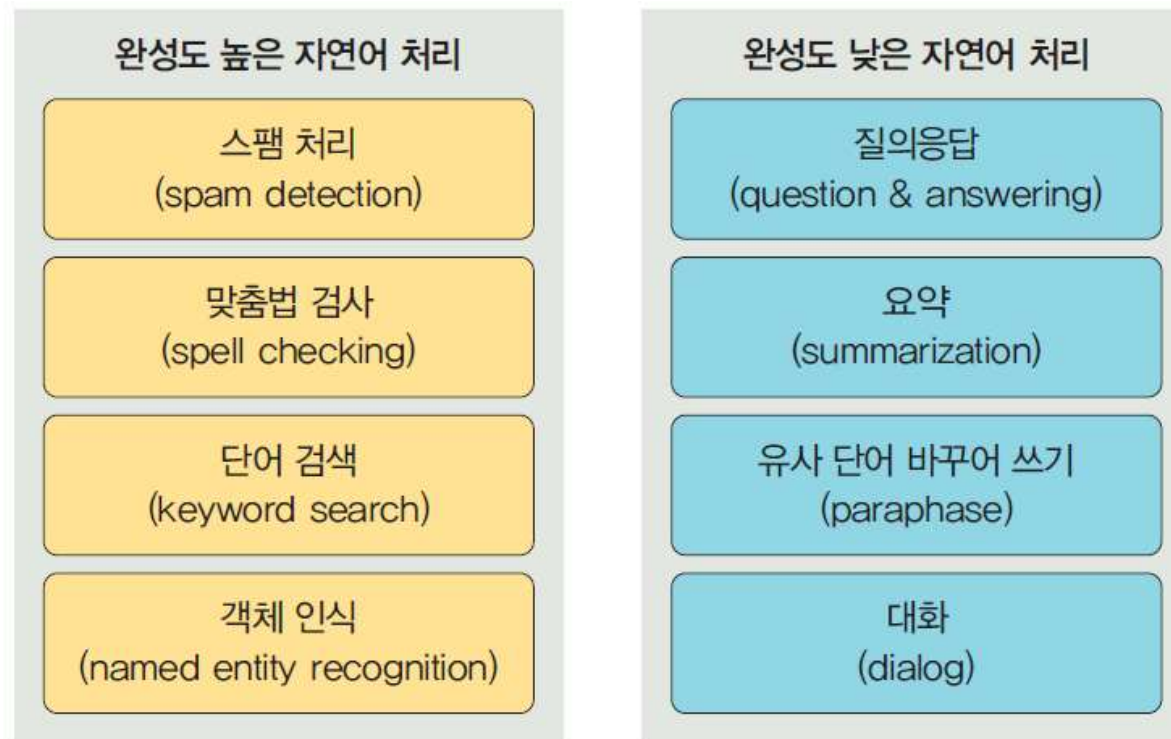
자연어 처리



● 자연어 처리

Robot Media Laboratory

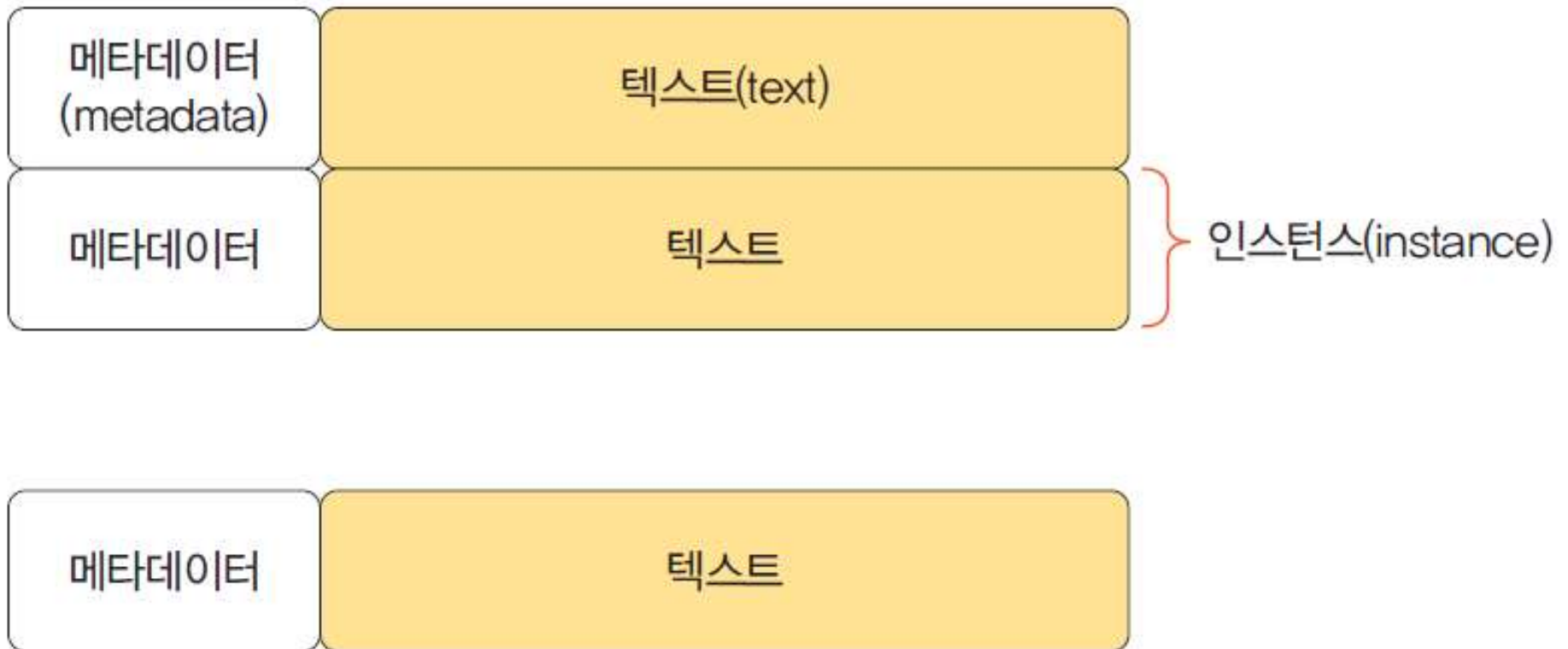
- 우리가 일상생활에서 사용하는 언어의 의미를 분석하여 컴퓨터가 처리할 수 있도록 하는 과정
- 언어 종류가 다르고 그 형태가 다양하기 때문에 처리가 매우 어려움



● 자연어 처리

Robot Media Laboratory

- 말뭉치(corpus(코퍼스)): 자연어 처리에서 모델을 학습시키기 위한 데이터이며, 자연어



● 자연어 처리

Robot Media Laboratory

● 토큰(token)

- 자연어 처리를 위한 문서는 작은 단위로 나누어야 하는데, 이때 문서를 나누는 단위가 토큰
- 문자열을 토큰으로 나누는 작업을 토큰 생성(tokenizing)이라고 하며, 문자열을 토큰으로
- 분리하는 함수를 토큰 생성 함수라고 함

● 토큰화(tokenization)

- 텍스트를 문장이나 단어로 분리하는 것을 의미
- 토큰화 단계를 마치면 텍스트가 단어 단위로 분리



● 자연어 처리

Robot Media Laboratory

● 불용어(stop words)

- 문장 내에서 많이 등장하는 단어
- 분석과 관계없으며, 자주 등장하는 빈도 때문에 성능에 영향을 미치므로 사전에 제거해 주어야 함

● 어간 추출(stemming)

- 단어를 기본 형태로 만드는 작업

consign
consigned
consigning
consignment

} consign

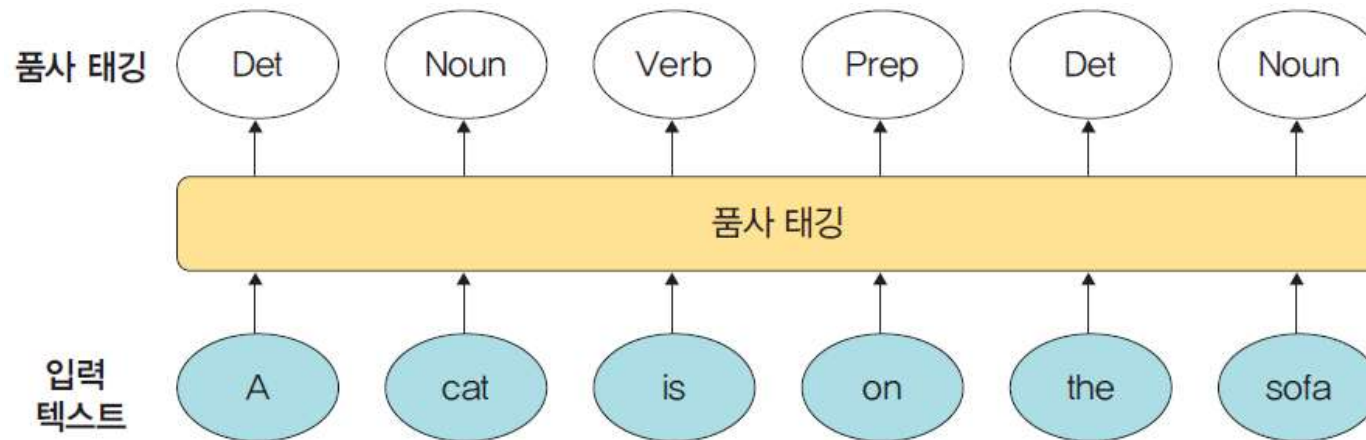


● 자연어 처리

Robot Media Laboratory

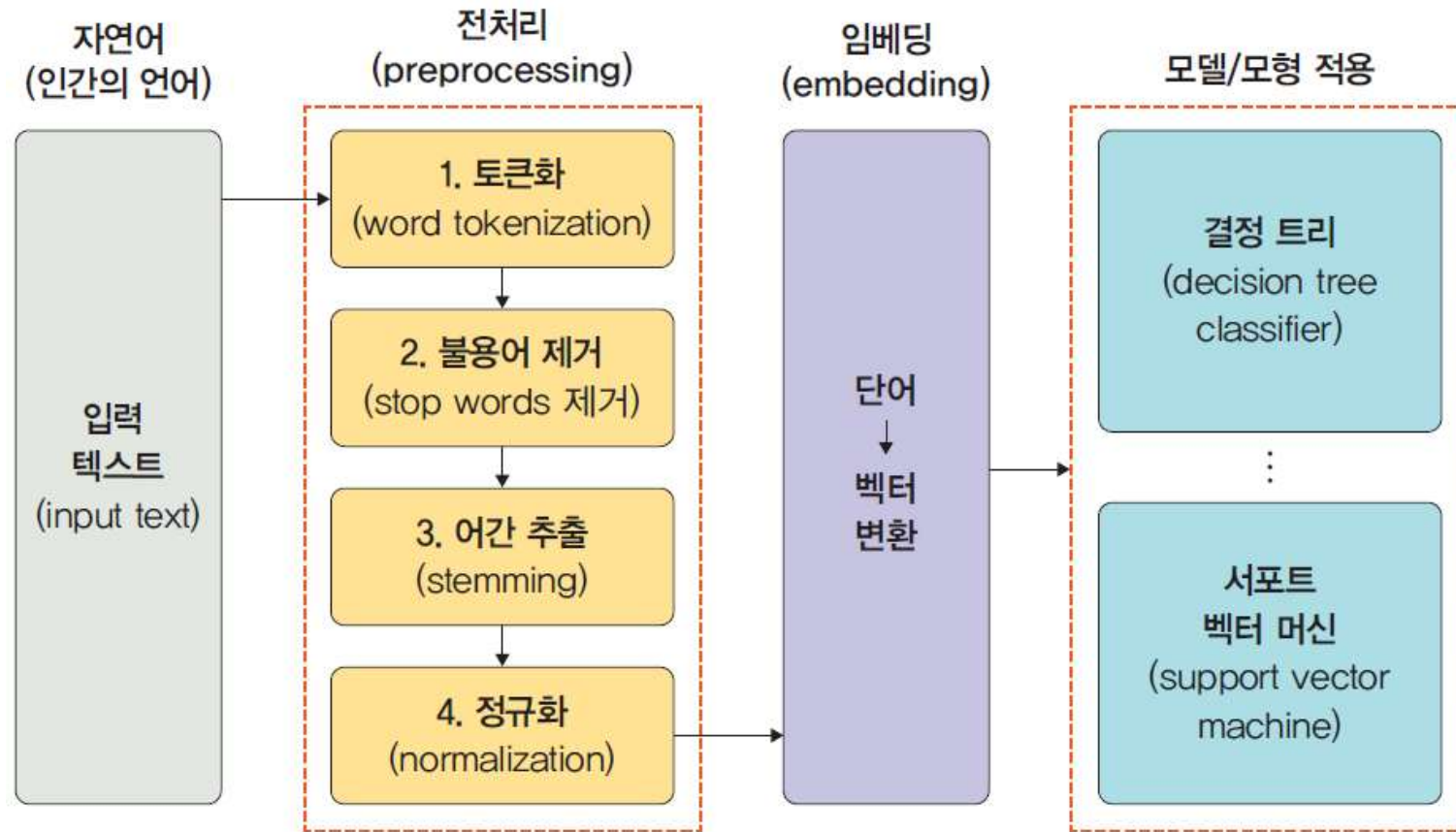
● 품사 태깅(part-of-speech tagging)

- 주어진 문장에서 품사를 식별하기 위해 붙여 주는 태그(식별 정보)를 의미



● 자연어 처리

Robot Media Laboratory

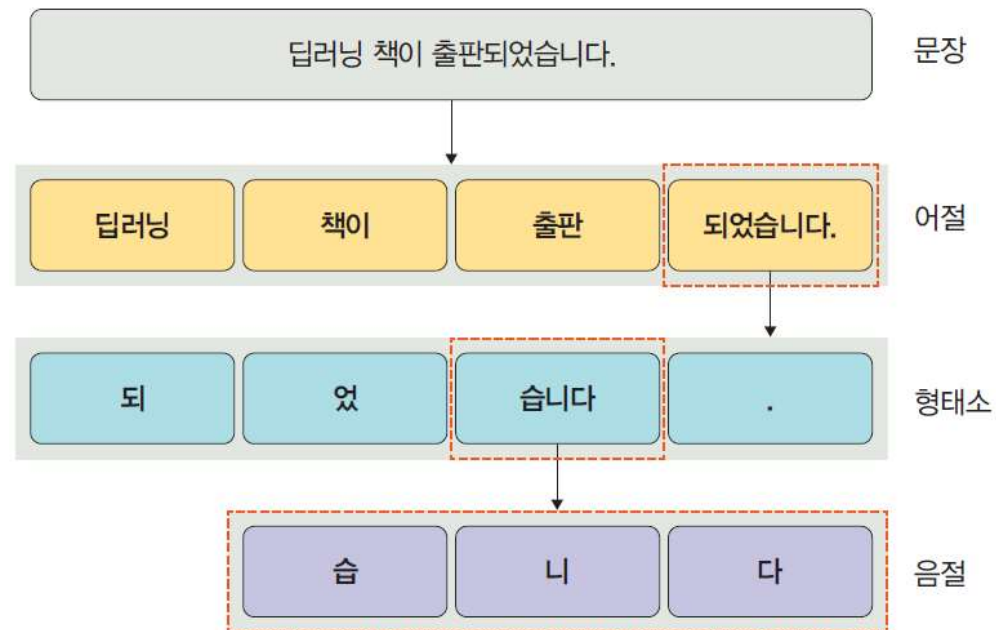


● 자연어 처리

Robot Media Laboratory

● 형태소

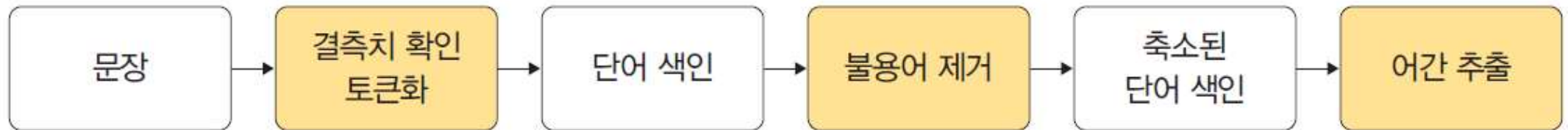
- 형태소는 언어를 쪼갤 때 의미를 가지는 최소 단위
- 다음 그림은 형태소 분석을 위한 단계를 도식화한 것



● 자연어 처리

Robot Media Laboratory

● 전처리



● 자연어 처리

Robot Media Laboratory

● 정규화(normalization)

- 표현 방법이 다른 단어들을 통합시켜서 같은 단어로 만들어 주는 것
- 머신 러닝/딥러닝은 데이터 특성들을 비교하여 패턴을 분석
- 값이 더 크면 큰영향을 가짐
- 중요한 것은 값이 크다고 해서 분석에 더 중요한 요소라고 간주 할 수 없기 때문에 정규화가 필요한 것

