20231121

오늘의 주제 : 생물학적 뉴런에서 인공 뉴런까지
- 케라스로 다층 퍼셉트로 구현하기
- 신경망 하이퍼파라미터 튜닝하기

10.인공 신경망

인공 신경망(Artificial Neural Network)
- 뇌에 있는 생물학적 뉴런의 네트워크에서 영감을 받은 머신러닝 모델
▪ 딥러닝의 핵심
▪ 강력하며 확장성이 좋음
▪ 인공신경망의 응용: 대규모 머신러닝 문제 다루기에 적합
▪ 구글 이미지: 수백만 개의 이미지 분류
▪ 애플의 시리: 음성인식 서비스
▪ 유튜브: 가장 좋은 비디오 추천
▪ 딥마인드의 알파고: 스스로 기보를 익히면서 학습
▪ 다층 퍼셉트론 (MLP, Multi-Layer Perceptron)

10.1 생물학적 뉴런에서 인공 뉴런까지

*인공 신경망 역사

-연결주의(Connectionism – 신경망 연구)

-1943년 명제논리를 사용해 동물 뇌의 생물학적 뉴런의 간단한 계산 모델을 제시

-1957년 : 프랭크 로센블래트 퍼셉트론 탄생 – 신경망 기반 인공지능 연구의 부흥기

-인공 신경망 첫번째 침체기
1969년 : 마빈 민스키 = 다층 퍼셉트론 XOR문제 O but 학습시킬 방법 X → 결론 : 퍼셉트론은 XOR문제에는 적용X
1970년대 이후 : R&D의 방향을 통계기술에 집중
1980년대 : 전문가 시스템이 도입 (통계기반)

-오류 역전파 알고리즘
1974년: 폴 웨어보스가 오류 역전파 알고리즘으로 다층 퍼셉트론을 학습시키는 방법을 박사 학위 논문으로 발표

1986년:제프리 힌튼 오류 역전파 알고리즘으로 다층 퍼셉트론을 학습시키는데 성공
1998년 얀 르쿤: CNN(Convolutional Neural Network) 발표

-1990년대 서포트 벡터 머신 같은 다른 강력한 머신러닝 기술 등장

-2000년대 중반 침체 : 그레이언트 소실 문제(기울기 없어져서 학습이 안됨)
→ 해결 : 시그모이드 대신 ReLU 함수 , 가중치 초기화 방법 개선

-2012년 ILSVRC의 우승은 알렉스넷(AlexNet)이라 명명된 CNN 기반의 인공 신경망이 차지

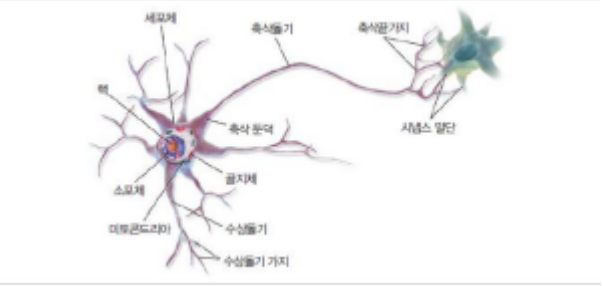
-현재 인공 신경망을 기반으로 한 놀라운 제품들이 출시됨.

인공 신경망의 부흥

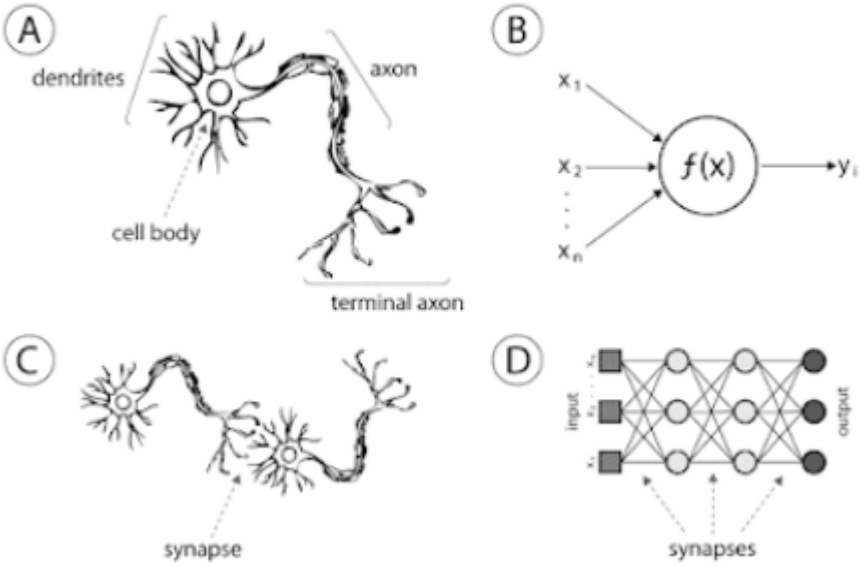
컴퓨터 성능 개선, 대량의 데이터, 알고리즘 개선

10.1.1 생물학적 뉴런

idea : 생물학적 뉴런



신경망 아이디어(



10.1.2 뉴런을 사용한 논리 연산

인공 뉴런
생물학적 뉴런에 착안한 매우 단순한 신경망 모델
인공신경망 : 논리 연산을 수행

간단한 인공 뉴런 모델로 인공 뉴런의 네트워크를 만듬.

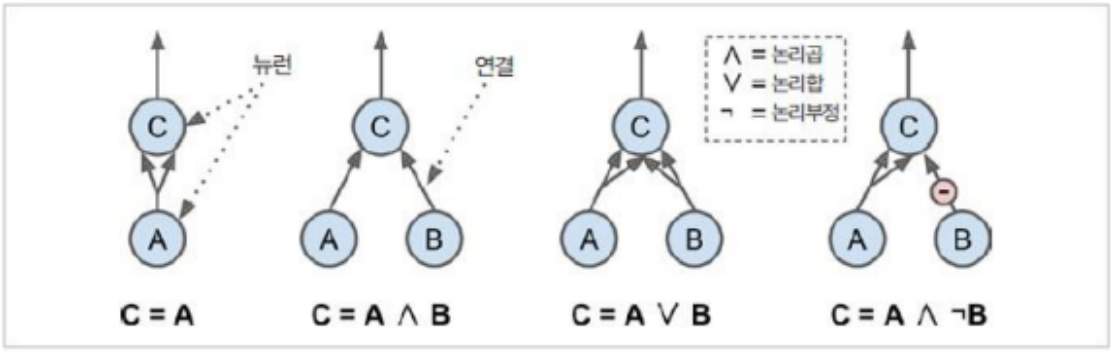


그림 10-3 간단한 논리 연산을 수행하는 인공 신경망

10.1.3 퍼셉트론 소개

*퍼셉트론
-가장 간단한 인공 신경망 구조

1957년 프랑크 로젠블라트

-TLU 또는 LTU(linear threshold unit) 인공 뉴런 활용
*TLU(threshold logic unit)
: 입력값과 가중치를 곱한 값들의 합에 계단함수(step function, 0과1밖에 없음) 적용

$$z = \mathbf{x}^T \mathbf{w}$$
$$= x_1 w_1 + x_2 w_2 + \cdots + x_n w_n$$

$$h_{\mathbf{w}}(\mathbf{x}) = step(z)$$

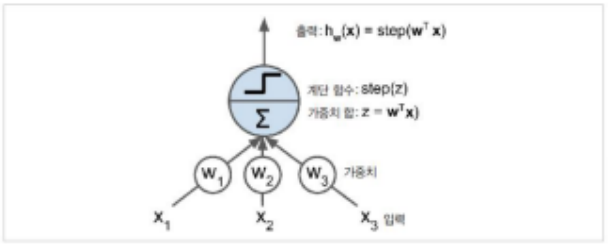


그림 10-4 TLU: 입력의 가중치 합을 계산한 다음 계단 함수를 적용하는 인공 뉴런

식 10-1 퍼셉트론에서 일반적으로 사용하는 계단 함수(임곗값을 0으로 가정)

$$\text{heaviside}(z) = \begin{cases} 0 & z < 0\text{일 때} \\ 1 & z \geq 0\text{일 때} \end{cases} \qquad \text{sgn}(z) = \begin{cases} -1 & z < 0\text{일 때} \\ 0 & z = 0\text{일 때} \\ +1 & z > 0\text{일 때} \end{cases}$$

8/67

10.1.3 퍼셉트론 (1)

*TLU와 선형 이진분류
- 하나의 TLU를 간단한 이진분류기로 활용 가능
- 모든 입력값(특성)의 선형 조합을 계산한 후에 임곗값을 기준으로 양성/음성 분류
- 작동은 로지스틱 회귀 또는 선형 SVM 분류기와 비슷.
- TLU 모델 학습 = 최적의 가중치 wi를 찾기
퍼셉트론 :

하나의 층에 여러 개의 TLU로 구성됨.
TLU 각각은 모든 입력과 연결됨.

입력 두 개와 출력 세 개로
구성된 퍼셉트론

10.1.3 퍼셉트론 (2)

퍼셉트론 학습 알고리즘
- 오차가 감소되도록 가중치 조절

$$w_{i,j}^{(\text{next step})} = w_{i,j} + \eta (y_j - \hat{y}_j) x_i$$

*퍼셉트론과 선형성
-각 출력 뉴런의 결정경계가 선형(계단함수때문임)
-복잡한 패턴 학습 못함. 퍼셉트론은 매우 단순한 경우만 해결 가능.

10.1.4 다층 퍼셉트론과 역전파

*다층 퍼셉트론(multilayer perceptron, MLP)
-퍼셉트론을 여러 개 쌓아올린 인공신경망
-입력층 하나, 은닉층(하나 이상의 TLU층),출력층으로 구성

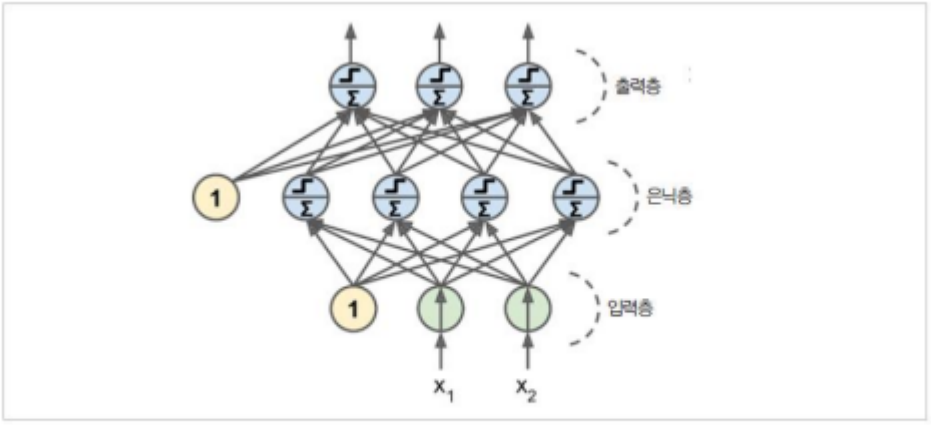
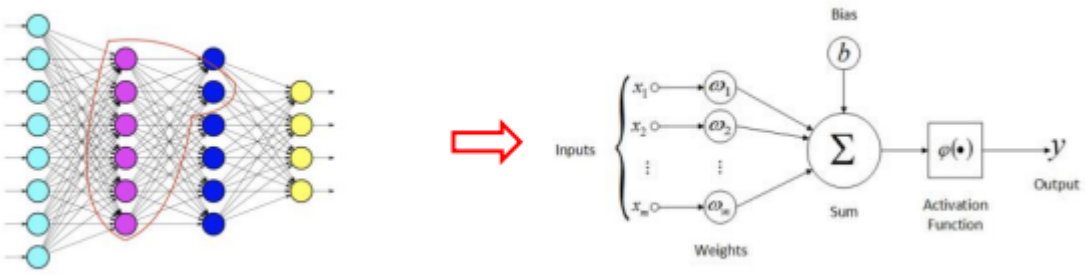
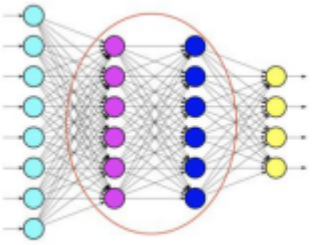


그림 10-7 입력층(입력 뉴런 2개), 은닉층(뉴런 4개), 출력층(출력 뉴런 3개)로 구성된 다층 퍼셉트론의 구조(여기에는 편향 뉴런을 표시했지만 보통 따로 나타내지 않습니다)

10.1.4 다층 퍼셉트론과 역전파 (1)
*완전연결 층(층에 속한 각각의 뉴런 이전 층의 모든 뉴런과 연결)의 출력 계산
-여러 개의 완전연렬 층으로 구성된 다층 퍼셉트론 모델 계산



하나의 층(세로)에서 이루어지는 입력과 출력을 행렬 수식으로 표현 가능



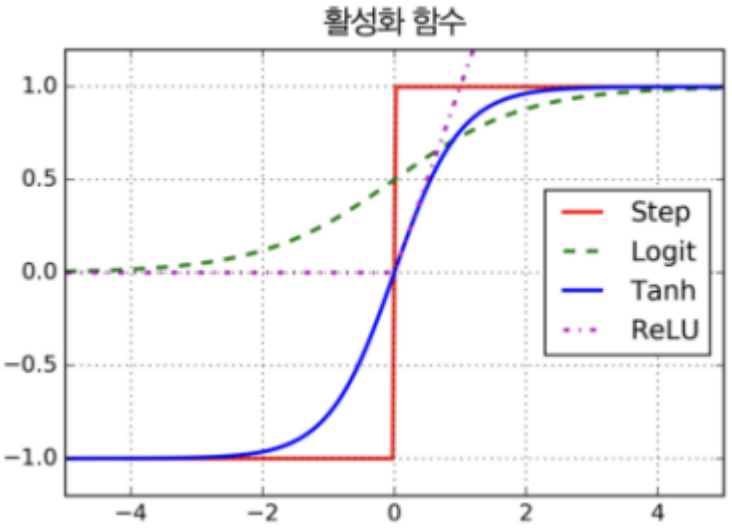
$$h_{\mathbf{w},\mathbf{b}}(\mathbf{X}) = \phi(\mathbf{XW} + \mathbf{b})$$

10.1.4 다층 퍼셉트론과 역전파 (2)

*심층신경망(DNN)
- 여러 개의 은닉층을 쌓아올린 인공신경망 .
*역전파 훈련 알고리즘
- 층이 많을 수록 훈련 과정이 어려워짐.
-1986년에 소개
-

1단계(정방향): 각 훈련 샘플에 대해 먼저 예측을 만든 후 오차 측정

2단계(역방향): 역방향으로 각 층을 거치면서 각 출력연결이 오차에 기여한 정도 측정
(미적분의 연쇄법칙(Chain rule) 적용)
3단계: 오차가 감소하도록 모든 가중치 조정
-MLP 특징
랜덤하게 설정함
활성화 함수를 계단함수 대신에 다른 함수(로지스틱,하이퍼볼릭 탄젠트 ,ReLU) 사용.



* 활성화 함수 대체 필요성
- 선형성을 벗어나기 위해
-복잡한 문제 해결 불가능
*비선형 활성화 함수를 충분히 많은 층에서 사용 = 매우 강력한 모델 학습 가능

10.1.5 회귀를 위한 다층 퍼셉트론

→ 요약 : 다층 퍼셉트론 회귀 가능

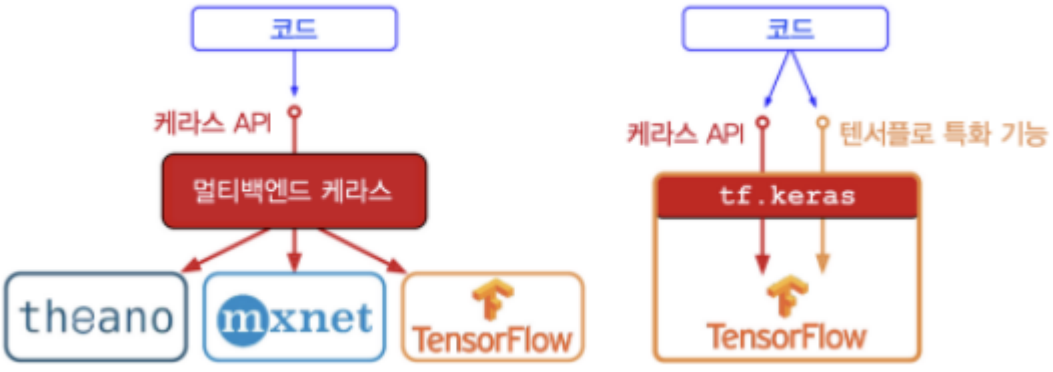
10.1.6 분류를 위한 다층 퍼셉트론

→요약 : 다층퍼셉트론으로 분류 가능
ex : 다중 클래스 분류
• MNIST 숫자 이미지. 0부터 9까지
• 출력층 활성화 함수: 소프트맥스 함수(정규화 시킴)

10.2 케라스로 다층 퍼셉트론 구현하기

*케라스:모든 종류의 신경망을 손쉽게 만들어 주는 최상위 딥러닝 API 제공

*종류 : 멀티 백엔드 케라스(구글의 텐서플로(Tensorflow), MS의 CNTK, Theano, 아파치 MXNet, 애플 Core ML), tensorflow.keras(텐서플로만 지원)



10.2.2 케라스 시퀀셜 API 활용한 이미지 분류

*패션 MNIST 활용

10.2.2(1) 케라스 시퀀셜 API 활용한 이미지 분류

*시퀀셜 API를 사용하여 모델 제작 : Sequential 클래스 내에 층을 쌓아 순차적 학습

- 은닉층: 2개

```
model = keras.models.Sequential()
model.add(keras.layers.Flatten(input_shape=[28, 28]))
model.add(keras.layers.Dense(300, activation="relu"))
model.add(keras.layers.Dense(100, activation="relu"))
model.add(keras.layers.Dense(10, activation="softmax"))
```

- Sequential 모델을 만들때 층의 리스트를 전달

```
model = keras.models.Sequential([
    keras.layers.Flatten(input_shape=[28, 28]),
    keras.layers.Dense(300, activation="relu"),
    keras.layers.Dense(100, activation="relu"),
    keras.layers.Dense(10, activation="softmax")
    ])
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
flatten (Flatten)            (None, 784)               0
_________________________________________________________________
dense (Dense)                (None, 300)               235500
_________________________________________________________________
dense_1 (Dense)              (None, 100)               30100
_________________________________________________________________
dense_2 (Dense)              (None, 10)                1010
=================================================================
Total params: 266,610
Trainable params: 266,610
Non-trainable params: 0
_________________________________________________________________
```

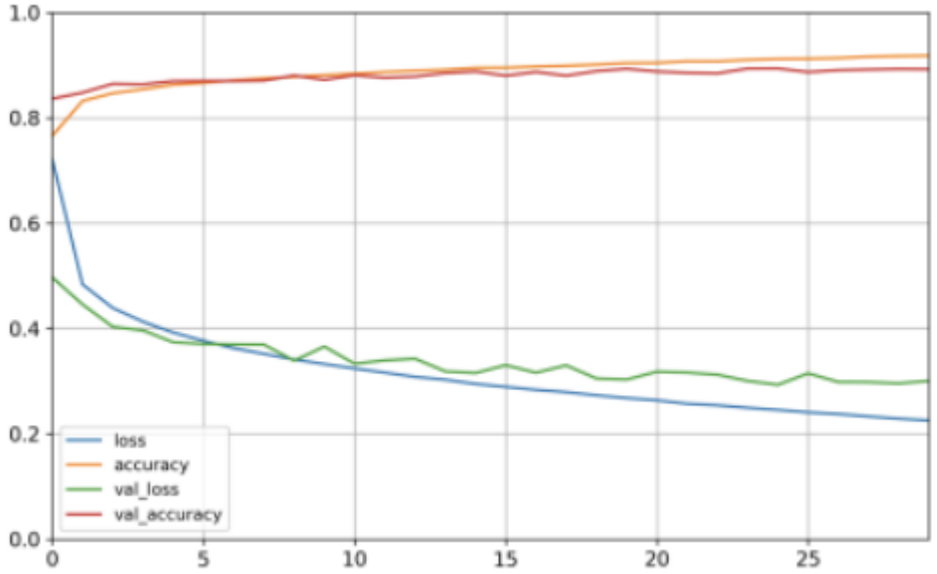
10.2.2(2) 케라스 시퀀셜 API 활용한 이미지 분류

*모델 컴파일 : 손실함수, 옵티마이저, 평가기준 등을 지정

*모델 훈련: fit() 메서드 호출

10.2.2(3) 케라스 시퀀셜 API 활용한 이미지 분류

*모델 학습 곡선
History 객체의 history 속성에 학습된 에포크의 손실과 정확도 기록



10.2.3(4) 케라스 시퀀셜 API 활용한 이미지 분류

*모델 평가 : evaluate() 메서드, 손실과 정확도 계산

*모델을 사용한 예측 :predict() 메서드로 예측

10.2.3 케라스 시퀀셜 API 활용 : 회귀

ex) 캘리포니아 주택가격 예측

→캘리포니아 데이터셋과 관련된 주의할 점
: 잡음이 많음(과대적합을 줄이기 위해 뉴런 수가 적은 하나의 은닉층만 사용)

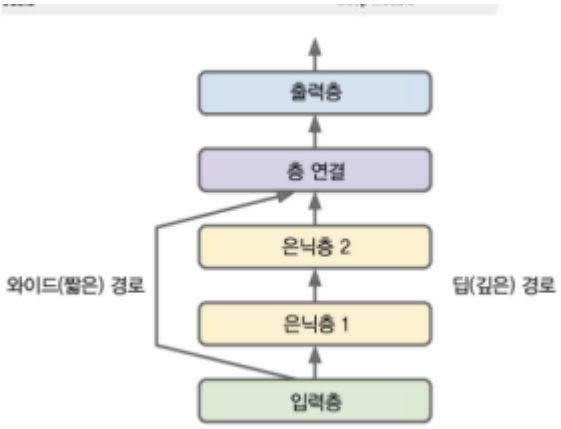
10.2.3(1) 케라스 시퀀셜 API 활용 : 회귀

케라스 Sequential 클래스 활용의 장단점
장점 : 사용하기 쉬움, 성능 우수.
단점 : 입출력이 여러 개 or 복잡한 네트워크를 구성 어려움

→Sequential 클래스 대신 함수형 API, 하위클래스(subclassing) API 등을 사용= 복잡, 강력한 딥러닝 모델 구축 가능

10.2.4 함수형 API를 사용해 복잡한 모델 만들기

*케라스 함수형 API 활용 : 모든 레이블을 순차적으로 처리하는 것 대신 다양한 신경망 구축을 위해 함수형 API 활용
*순차적이지 않는 신경망 ex: 와이드&딥(Wide & Deep) 신경망 → 깊게 쌓은 층을 사용, 복잡한 패턴/ 짧은 경로를 사용한 간단한 규칙 모두 학습가능한 모델



10.2.4(1) 함수형 API를 사용해 복잡한 모델 만들기

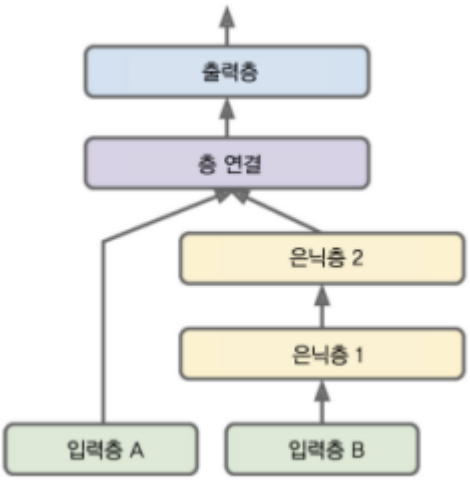
*와이드&딥 신경망의 활용 방법1
→캘리포니아주 주택가격 데이터셋 활용

inputA - hidden2로 연결

inputB- 순차적으로 연결 hid1 hid2 ..

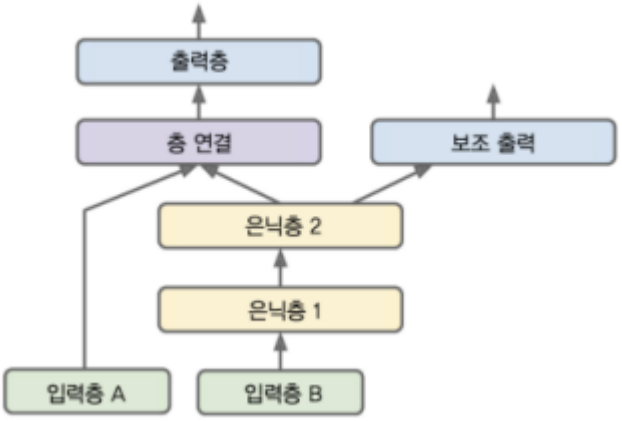
10.2.4(2) 함수형 API를 사용해 복잡한 모델 만들기

와이드&딥 신경망의 활용 방법2
다중 입력 사용 - 특성을 나누어 짧은 경로와 깊은 경로에 (중복을 허용하여) 특성을 나누어 보낼 수도 있음.
input_A: 5개의 특성 입력받음 O
input_B: 6개의 특성 입력받음 O
입력 뉴런마다 입력값 지정하여 학습해야 함



10.2.4(3) 함수형 API를 사용해 복잡한 모델 만들기

다중 출력이 필요한 경우
*여러 출력이 필요한 작업: 동일한 데이터에서 독립적인 여러 작업 수행(출력A 사람의 얼굴 표정 분류, 출력B 안경 썼는지 분류)



10.2.5 서브클래싱 API로 동적 모델 만들기
*Sequential 클래스와 함수형 API = 선언적 방식= 정적
-한 번 선언, 변경할 수 없는 모델 생성
-모델 저장, 복사, 공유 용이
-모델 구조 출력 및 모델 분석 용이
-반복문, 조건문 으로 동적 모델을 생성 시 명령형 프로그래밍 방식 요구됨
*서브클래스 API을 활용하여 동적 모델 생성 가능

-초기설정 메서드(**init**())를 이용하여 은닉층과 출력층 설정
-call() 메서드를 이용하여 층을 동적으로 구성 가능

10.2.6 모델 저장과 복원

▪ Sequential 모델과 함수형 API 를 사용해서 훈련된 모델 저장
-케라스= HDF5 포맷 이용, 모델 구조와 층의 모든 파라미터을 저장
-옵티마이저도 저장 (하이퍼파라미터와 현재 상태를 포함)

10.2.7 콜백 사용하기

*체크포인트(checkpoint) 저장시 사용
:대규모 데이터셋에서 훈련 시에 훈련 도중 일정 간격으로 체크포인트를 저장해야 함.

10.2.7 (1) 콜백 사용하기

*콜백함수 활용 – 조기 종료 구현

*사용자정의 콜백함수

10.2.8 텐서보드를 사용해 시각화하기

*텐서보드 기능
-훈련하는 동안의 학습곡선, 여러 실행 간의 학습곡선 비교, 계산그래프 시각화
-훈련통계 분석, 3D에 사영된 복잡한 다차원 이미지 시각화,

10.3 신경망 하이퍼파라미터 튜닝하기

신경망은 조정해야할 하이퍼파라미터가 매우 많음.
-은닉층,유닛수, 층, 뉴런, 학습률, 활성화 함수, 가중치 초기화 등

10.3 신경망 하이퍼파라미터 튜닝하기 (1)

*랜덤탐색 기법은 매우 오래 걸림.
→하이퍼파라미터 최적화에 사용할 수 있는 다양한 라이브러리 개발됨.
• Hyperopt Hyperas, kopt, Talos Keras Tuner
• Scikit-Optimize(skopt), Spearmint, Hyperband, Sklearn-Deap