인공지능

1장 딥러닝이란 무엇인가?

1.1 인공 지능과 머신 러닝, 딥러닝

인공지능: 사람이 지능적으로 수행하는 작업을 자동화하기 위한 연구

머신러닝: 명시적인 프로그래밍이 아닌 훈련(학습)을 통한 자동화

딥러닝: 입력 데이터를 기반으로 기대 출력에 가깝게 만드는 표현을 학습하는 것!

- 딥: 연속된 층으로 머신 러닝(신경망)

학습: 입력을 정확한 타깃에 매핑하기 위해 weight를 찾는 과정

손실함수: 예측값과 타깃을 비교하여 손실 값을 계산하는 함수

옵티마이저: loss값을 역전파 알고리즘을 통해 가중치에 반영

1.2 딥러닝 이전: 머신 러닝의 간략한 역사

확률적 모델링(나이브 베이즈 알고리즘, 로지스틱 회귀(분류알고리즘임))

초창기 합성곱 신경망(LeNet) -> 손글씨 숫자 이미지 분류

커널 방법(SVM) - 간단한 분류 문제

딥러닝은 특성 공학 부분을 자동화 -> 더 복잡한 데이터도 가능해짐

1.3 왜 딥러닝일까? 지금일까?

알고리즘이나 기술은 이미 20년전

지금 급부상하는 이유

1 하드웨어의 발전

2 인터넷의 발전으로 데이터셋 수집 용이

3 알고리즘 강화(활성함수, 옵티마이저 등)

2장 시작하기 전에: 신경망의 수학적 구성 요소

2.1 신경망과의 첫 만남

MNIST 예제(흑백 손글씨 숫자 이미지 분류)

6만개의 train, 1만개의 test 데이터셋

\* dense: 입출력 모두 연결

\* softmax: 10개의 확률 점수가 들어있는 배열(모두 더하면 1)

과대적합overfitting: 모델이 훈련 데이터보다 테스트 데이터에서 성능이 낮아지는 경향

2.2 신경망을 위한 데이터 표현

텐서: 다차원 numpy 배열(데이터를 위한 컨테이너)

- 0D텐서(스칼라): 숫자

- 1D텐서(벡터): 배열

- 2D텐서(행렬)

- 3D~ 고차원: 이미지 동영상

2.3 신경망의 톱니바퀴: 텐서 연산

2.4 신경망의 엔진: 그래디언트 기반 최적화

- 신경망의 모든 연산 미분 가능

- 그래디언트 계산이 훨씬 좋은 방법

확률적 경사 하강법(SGD): 배치 데이터를 무작위로 선택

- step값이 너무 작으면 지역 최솟값에 갇힐 수 있음

- 너무 크면 gradient vanishing

- 모멘텀(과거에 업데이트된 가중치를 고려)을 사용하여 수렴 속도, 지역 최솟값 해결

2.5 첫 번째 예제 다시 살펴보기

3장 신경망 시작하기

3.1 신경망의 구조

dense(밀집, 완전 연결) : 간단한 벡터데이터

LSTM: 3D 시퀀스 데이터

Conv2D: 4D텐서

3.2 케라스 소개

케라스 특징

1 CPU, GPU 모두 가능

2 다양한 API, 라이브러리

3 어떤 딥러닝 모델에도 적합

4. MIT 라이센스

5 텐서플로우 라이브러리 사용 가능

3.3 딥러닝 컴퓨터 셋팅

3.4 영화 리뷰 분류: 이진 분류 예제

IMDB: 2만5천개 훈련데이터 / 2만5천개 테스트데이터

리뷰의 단어들을 리스트로 만들고, 원-핫 인코딩하여 vectorize(모델에 주입할 수 있는 데이터 형식으로 만들기위해)

relu를 사용하는 이유: 선형 연산만으로는 층을 쌓는 이점이 없기 때문

model.fit()을 통해 학습을 진행하면 history 객체 반환(loss값과 metrics값 반환 + train/val따로)

3.5 뉴스 기사 분류: 다중 분류 예제

reuters 로이터 데이터셋: 단일 레이블 다중 분류

import to\_categorical

to\_categorical(train\_labels) 함수사용하면 원핫 인코딩 간편하게

16개의 노드로 46개의 클래스를 구분하기엔 제약이 많음 (정보의 병목)

-> 64개로 노드로 확장

-> 충분히 큰 중간층을 두어야 하는 이유

model.predict(x\_test) 로 테스트

3.6 주택 가격 예측: 회귀 문제

특성마다 스케일이 다르므로 데이터 정규화(-평균/std) -> 0근처로 맞춰지게함

샘플 개수가 적으므로 2개 층으로만 작게 구성(과대적합 방지)

k-fold (np.concatenate() 사용)

스케일이 많이 다른 첫 10개의 데이터포인트를 제외

부드러운 곡선을 위해 지수이동평균 사용

5epochs 이동평균

4장 머신 러닝의 기본 요소

4.1 머신 러닝의 네 가지 분류

1 지도학습: 타깃에 입력 데이터를 매핑

2 비지도학습: 타깃 사용하지않고 입력 데이터에 대한 흥미로운 변환 찾음 (데이터 분석) / 차원축소, 군집

3 자기 지도 학습: 지도학습이지만 사람이 만든 레이블 사용하지 않음(오토 인코더)

4 강화 학습: 딥마인드 / 보상을 최대화하는 행동

4.2 머신 러닝 모델 평가

머신 러닝의 목표는 잘 작동하는 '일반화'된 모델을 얻는 것

1 훈련, 검증, 테스트 세트로 나누기

검증세트에 기반하여 하이퍼파라미터 조절하면 정보 누설! (테스트 세트로 학습하면 안되는 이유)

2 단순 홀드아웃 검증, k-fold 검증, 셔플링

3 다음 사항 유념(대표성 있는 데이터X, 시계열에서 셔플X, 데이터 중복X)

4.3 데이터 전처리, 특성 공학, 특성 학습

데이터 전처리(벡터화)(원핫인코딩)

정규화

누락된 값 다루기(0이 중요한 의미일경우 무의미한 값으로 0 사용을 지양)

특성공학 : 학습이아닌 하드코딩을 통해 데이터 가공

- 적은 자원 사용, 더 적은 데이터일 경우 도움됨

4.4 과대적합과 과소적합

최적화: loss값 줄이기

일반화:overfitting 없애기

과소적합된상태면 더 학습할 여지가 있는 것

규제

1 네트워크 크기 축소: 네트워크 줄이면 overfitting 방지

2 가중치 규제(L1, L2)

- L1: 절댓값 -> 특성 공학에서 사용

- L2: 제곱값 -> 일반화에 초점

3 드롭 아웃: 무작위로 일부 특성을 제외 -> 중요하지 않은 우연한 패턴을 깨트림!

4.5 보편적인 머신 러닝 작업 흐름

1 문제 정의(뭘 예측할건지, 문제의 종류가 무엇인지)

2 성공 지표 선택

3 평가 방법 선택

4 데이터 준비

5 활성함수, 손실함수, 최적화 설정

6 과대적합 모델 만들기

7 규제, 튜닝 통해 과대적합 없애기