

파이썬 기반의 MNIST 숫자 인식 프로그램 분석 보고서

인공지능 기술 (Python 프로그래밍)

202578218 정지은 AISW계열

목차

MNIST 숫자 인식 개요.....	1
MNIST 숫자 인식 순서 및 방법.....	2
1. 데이터셋 로드 및 분할.....	2
2. 모델 정의 생성.....	2
- Flatten Layer.....	2
- Dense Layer.....	2
3. 모델 컴파일 및 훈련.....	2
- batch.....	2
- epoch.....	2
“에포크가 계속될수록 loss가 줄어드는 쪽으로, 정확도가 증가하는 쪽으로 최적 모델을 찾아간다.”.....	3
4. 모델 평가 및 검증.....	4
로지스틱 회귀와 시그모이드 함수.....	4
로지스틱 회귀.....	4
시그모이드 함수.....	4
결론.....	5

MNIST 숫자 인식 개요

MNIST 숫자 인식 프로그램은 대표적인 파이썬 기반의 인공지능 모델로서, 손으로 쓴 숫자 0-9를 컴퓨터가 자동으로 인식하게 만드는 이미지 분류 모델이다. MNIST(Modified National Institute of Standards and Technology database)는 0-9 숫자 이미지와 해당 이미지에 대한 숫자 라벨이 포함된 데이터 베이스이다. 약 7만 개의 훈련용 이미지와 1만개의 테스트 이미지를 포함한다. 각 이미지는 회색조로 되어 있거 28x28 픽셀로 구성됐고 0-255 사이의 숫자 행렬로 표현되어 있다.

이 보고서는 텐서플로우 등의 다양한 오픈소스 자료들과 chat-GPT의 설명을 통해 MNIST 숫자 인식 프로그래밍에 관해 공부한 내용을 정리하여 담았다.

MNIST 숫자 인식 순서 및 방법

1. 데이터셋 로드 및 분할

- 훈련 집합 e.g) `x_train, y_train`
- 테스트 집합 e.g) `x_test, y_test`
- `train`과 `test` 데이터 분할은 8:2 비율로 분할

다운로드 받은 데이터들을 불러 훈련 집합과 테스트 집합으로 분할한다. 8:2 비율을 고수하는 이유를 챗지피티에게 물어, 일반화 성능 평가, 과적합 방지, 통계적 안정성 확보라는 답변을 받았다. 여기서 프로그램은 결국 안정성을 우선시하고 우선시킬 수 밖에 없다는 것을 다시 한 번 깨달을 수 있었다. 환경이나 랜덤한 상황에 따라 성능이 달라지는 모델은 수요가 없을 수 밖에 없다. 코드를 짤 때에 일단 과제를 해결하는 것에만 급급하여 문제를 풀었었는데, 효율성과 안정성에 관한 고려를 꼭 해야겠다고 다짐하는 계기가 되었다.

2. 모델 정의 생성

- Flatten Layer

다차원 배열을 1차원 벡터로 펼치는 역할을 하는 인공신경망의 기본 층으로서 Dense계층에 연결하기 위해서는 필수적이다.

- Dense Layer

모든 뉴런이 이전 층의 모든 뉴런과 연결된 층으로, 완전 연결층이라고도 한다. 입력 벡터와 가중치 행렬을 곱한 뒤 활성화 함수로 출력하는 구조를 가진다. 이미지의 위치 기반 특성을 처리한 후, 최종적으로 분류 또는 회귀 등의 결과를 도출하기 위한 목적이다.

선형 모형 $y = wx + b$ 형태의 레이어

3. 모델 컴파일 및 훈련

- batch

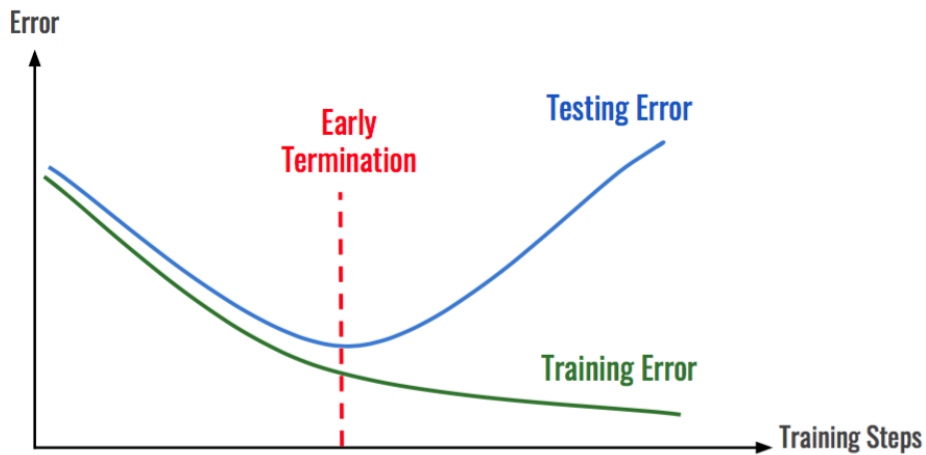
전체 학습 데이터셋 중에서 한 번의 파라미터 업데이트에 사용되는 소규모 묶음이다.
(기본 32개)

- epoch

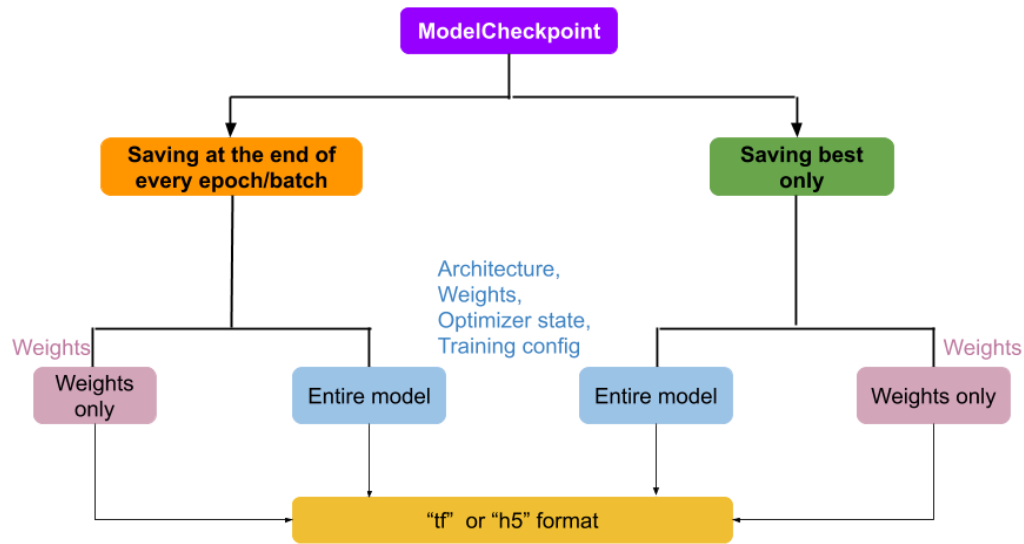
전체 훈련 데이터셋을 한 번 사용하여 학습이 완료된 단위로, 전체 데이터를 한 번 처음부터 끝까지 훑는 과정까지 포괄한다.

“에포크가 계속될수록 **loss**가 줄어드는 쪽으로, 정확도가 증가하는 쪽으로 최적 모델을 찾아간다.”

Gradient Descent 반복 적용되고, 경사하강법은 반복할수록 더 좋은 방향으로 수렴하기 때문이다. 그러나 예외적으로 **epoch**가 지나치게 많을 경우, 과적합 **Overfitting**이 일어날 수 있다. 때문에 손실값이 더이상 감소하지 않으면 조기 종료하는 **EarlyStopping** 기법이나 성능이 가장 좋았던 시점의 모델을 저장하는 **ModelCheckpoint** 기법을 사용하여 과적합 현상을 막아야 한다.



Early Stopping (Medium)



ModelCheckpoint (Medium)

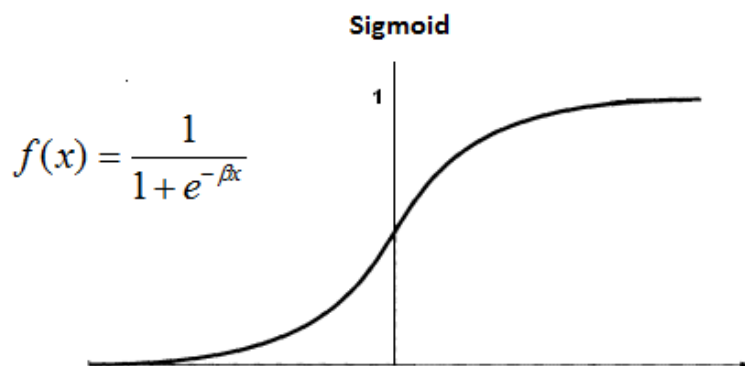
4. 모델 평가 및 검증

시연을 통해 모델을 평가하여 성능을 검증할 수 있다.

로지스틱 회귀와 시그모이드 함수

로지스틱 회귀

로지스틱 회귀는 딥러닝의 기초가 되는 이진 분류모델로, 선형 회귀와 다르게 출력값을 확률로 해석할 수 있게 만든 모델이다. 출력은 항상 0과 1 사이의 실수값이 된다.



(wikidocs)

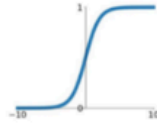
시그모이드 함수

로지스틱 회귀와 신경망에서 출력값을 확률처럼 제한하는 비선형 함수이다. 분류 문제에서 예측값을 확률로 제한하기 위하여 사용된다.

Activation Functions

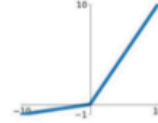
Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



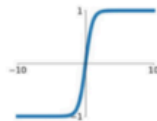
Leaky ReLU

$$\max(0.1x, x)$$



tanh

$$\tanh(x)$$

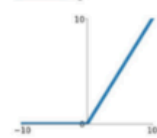


Maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

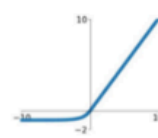
ReLU

$$\max(0, x)$$



ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



Different Activation Functions and their Graphs

(Medium)

결론

파이썬 기반의 인공지능 모델인 MNIST 숫자 인식 프로그램에 대해 공부하며 인공지능 모델이 어떻게 구현되는지에 대한 과정을 전체적으로 이해할 수 있었다. 복잡한 함수 체계나 낯선 개념들이 이해가 잘 되지는 않지만, 그럼에도 불구하고 이미 구현된 오픈소스를 훑어보며 직접 실행까지 할 수 있었던 것이 전반적인 이해도를 높이는 데에 많은 도움이 되었다. 특히 저번에 감정 인식 프로그램에 도전하다가 최종적으로 성공하지는 못했는데 다시 도전하고 싶은 흥미가 생길 수 있었다.