DLPyTorch CH.8

8.1 성능 최적화

8.1.1 데이터를 사용한 성능 최적화

- 최대한 많은 데이터 수집
- 데이터 생성
- 데이터 범위(scale) 조정
  - 시그모이드 - 범위: 0~1
  - 하이퍼볼릭 탄젠트 - 범위: -1~1
- 정규화, 규제화, 표준화

8.1.2 알고리즘을 이용한 성능 최적화

- 최적의 알고리즘을 찾기 위해서 여러 알고리즘을 선택해 모델을 훈련시켜 보고 최적의 성능을 보이는 알고리즘을 선택

8.1.3 알고리즘 튜닝을 위한 최적 성능화

- 진단: 성능 향상이 어느 순간 멈추었다면 원인을 분석해야함. 모델에 대한 평가 결과를 바탕으로 성능 향상의 원인을 분석
- train 성능이 test 성능보다 눈에 띄게 좋으면 overfitting 의심해야함. 이를 해결하기 이해 규제화 진행
- train, test 가 모두 성능이 좋지 않다고 underfitting 의심. 이를 해결하기 위해 네트워크 구조를 변경하거나 train을 늘리기 위해 epoch 수를 조정해야함.

- train 성능이 test 성능을 넘어서는 변곡점이 있다면 조기 종료를 고려할 수 있습니다.

- 가중치
  - 초깃값은 작은 난수를 사용. 작은 난수라는 숫자가 애매하다면 오토인코더와 같은 비지도 학습을 이용해 사전 훈련을 진행할 수 있음.
- 학습률
  - 초기에 난수 선택 후 조금씩 변경. 네트워크의 계층이 많다면 학습률은 높아야 하고 계층이 적다면 학습률은 작게 설정되어야 함.
- 활성화 함수
  - 활성화 함수를 변경할 때 손실 함수도 함께 변경해야하는 경우가 많아 신중하게 선택해야함.
- 배치/에포크
  - 큰 에포크와 작은 배치를 사용하는 것이 딥러닝의 트렌드
- 옵티마이저
  - adam, RMSDrop
- 네트워크 구성
  - 구성 방식에 대한 고민

8.1.4 앙상블을 이용한 성능 최적화

8.2 하드웨어를 이용한 성능 최적화

8.2.1 CPU와 GPU 사용의 차이

- CPU는 한번에 하나의 명령어만 처리
- GPU는 병렬적으로 처리.

8.3 하이퍼파라미터를 이용한 성능 최적화

8.3.1 배치 정규화를 이용한 성능 최적화

- 정규화
- 규제화
- 표준화
- 배치 정규화
    - 기울기 소멸 문제와 기울기 폭발 문제 해결
    - 분산된 분포를 정규분포로 만들기 위해 표준화와 유사한 방식을 미니 배치에 적용하여 평균은 0으로, 표준편차는 1로 유지하도록함.

8.3.2 드롭아웃을 이용한 성능 최적화

- 일정 비율의 뉴런만 사용하고 나머지 뉴런에 해당하는 가중치는 업데이트하지 않는 방식

8.3.3 조기 종료를 이용한 성능 최적화