

**딥러닝 hw2 과제**

**과목명 딥러닝**

**담당교수 장익범교수님**

**제출일 20231015**

**전공 컴퓨터전자시스템**

**학번 201904458**

**이름 이준용**

**1. Learning curve (y축: Train\_loss & Validation\_loss curve, x축: Epochs)**

스크린샷, 텍스트, 라인, 그래프이(가) 표시된 사진

자동 생성된 설명

**2. Hyperparameter**

* 에폭(epoch) 수 => 10

: 데이터셋을 반복하는 횟수

Epoch은 batch size와 learning rate을 각각 수치를 조정해가면서 loss curve를 확인했는데 그래프가 epoch 수가 5이상부터는 일정하게 감소하는 형태를 띄어서 10으로 계속 고정하여 훈련시켰습니다.

* 배치 크기(batch size) => 32

: 매개변수가 갱신되기 전 신경망을 통해 전파된 데이터 샘플의 수

Batch size를 32와 64를 둘 다 훈련시켰을 때 32인 경우에 더 훈련이 잘되어 32로선택했습니다.

* 학습률(learning rate) =>0.00001

: 각 batch/epoch에서 모델의 매개변수를 조절하는 비율. 값이 작을수록 학습 속도가 느려지고, 값이 크면 학습 중 예측할 수 없는 동작이 발생할 수 있습니다.

처음 훈련시켰을 때 0.001로 훈련시켰는데 예측할 수 없는 loss로 “nan”이 출력되었습니다.

그래서 0.1씩 곱해가면서 learning rate을 줄여갔습니다. 200\*200\*3= 120000 정도의 feature인데 learning rate도 1e-5정도 되어야 fitting이 잘 될 것이라고 생각했습니다.

**3. 예측에 가장 중요한 feature를 순서대로 1,000가지 나열하기.**

텍스트, 스크린샷, 그래프, 라인이(가) 표시된 사진

자동 생성된 설명

**4. 위의 Feature Importance Map으로부터 깨달은 사실 혹은 배운 점에 대해 서술**

Feature Importance Map은 머신러닝 모델이 예측을 수행하는 데 있어 각 피처가 얼마나 중요한 역할을 하는지를 시각화한 도구입니다. 이를 통해, 다음과 같은 사실들을 깨달을 수 있습니다:

피처의 상대적 중요성: Feature Importance Map은 각 피처가 모델의 예측에 어느 정도 기여하는지 보여줍니다. 이를 통해 가장 중요한 피처들이 무엇인지, 그리고 그것들이 어떻게 타겟 변수와 관련되어 있는지 이해할 수 있습니다.

모델의 해석 가능성: 복잡한 머신러닝 모델, 특히 앙상블 모델과 신경망 등은 종종 "블랙 박스"로 비유됩니다. Feature Importance Map은 이러한 복잡한 모델의 내부 작동 방식에 대해 좀 더 많은 인사이트를 제공하며, 모델이 왜 특정 예측을 만드는지 설명하는데 도움이 됩니다.

특징 선택: Feature Importance Map으로부터 가장 영향력 있는 변수들을 파악하고, 이 정보를 바탕으로 feature selection 혹은 feature engineering을 진행할 수 있습니다. 즉, 중요도가 낮거나 영향력 없는 변수들을 제거함으로써 계산 비용을 줄일 수 있고, 과적합(overfitting) 문제를 완화할 수 있습니다.

데이터 문제 식별: 만약 예상치 못하게 어떤 변수가 너무 큰 영향력을 가진다면 데이터에 문제가 있는 것일 가능성도 확인해볼 필요가 있습니다. 예를 들어 데이터 유출(leakage)나 전처리 과정에서 발생한 오류 등입니다.

따라서 Feature Importance Map는 데이터 과학자와 분석가들에게 매우 유용한 도구로 작용합니다.

**5. 과제를 진행하며 어려웠던 점 서술.**

얼굴 사진을 통한 나이 예측 모델 학습 관련 경험이 없다 보니 Hyperparameter 값을 이렇게 저렇게 해보면서 시간을 많이 소모했습니다.

**6. 과제를 진행하며 배운 점과 느낀 점을 서술**

데이터를 모델에 입력하기 전에 적절한 전처리 과정을 거쳐야 합니다. 이미지 크기 조정, 텐서로 변환, 정규화 등의 과정을 통해 데이터를 모델에 맞게 가공할 수 있습니다.

다음으로 PyTorch를 사용하여 모델을 처음부터 구성하는 방법을 배웠고 터득했습니다. nn.Module 클래스를 상속받아 필요한 레이어들을 정의하고, forward 메서드에서 입력과 출력의 관계를 정의합니다.

학습 및 검증: 훈련 데이터셋과 검증 데이터셋으로 나누어 모델을 학습시키고 검증하는 방법을 익혔습니다. 에폭마다 손실(loss) 값을 계산하고 기록하여 학습 과정을 시각화할 수 있었습니다.

CUDA 활용: CUDA가 사용 가능한 경우 GPU를 활용하여 모델 및 데이터를 GPU로 이동시켜 연산 속도를 향상시킬 수 있었습니다.

Learning Curve 그리기: 학습 및 검증 손실(loss) 값을 기록하고 이를 시각화하여 Learning Curve(학습 곡선)을 그릴 수 있었습니다. 이는 모델의 학습 진행 상황과 성능 변화를 파악하는 데 도움이 되었습니다.

Feature Importance Map 분석: 모델 내부에서 각 feature들의 중요도에 대한 분석이 가능함을 배웠습니다. Feature Importance Map은 해당 feature가 예측 결과에 얼마나 영향력이 있는지 알려줍니다.

요약 정보 출력: torchsummary 패키지를 사용하여 모델의 요약 정보(summary)와 파라미터 개수 등 중요한 정보들을 확인할 수 있었습니다.

위와 같은 내용으로 과제 진행하며 배운 점은 다양한 딥러닝 작업 단계에서 필요한 기본적인 개념과 코드 작성 방법들입니다. 또한 실제로 코드 실행 및 결과 확인하는 경험도 얻게 되어 실전적인 역량 강화에 도움이 되었습니다.

이번 과제를 차근차근 해보면서 다음 딥러닝 관련 주제로 스스로 훈련할 때는 이번 과제가 밑거름이 될 것입니다.