회고록

MLP

처음에 배운 MLP 이론에서는 기존에 학과에서 들은 인공지능 수업을 통해 알고 있었던 개념이라 생각했던 손실함수와 비용함수에 대해 차이점에 대해 모르고 있었다는 것을 정리하면서 알게 됐다.

새롭게 배우게 된 것은 케라스를 통해 모델을 만드는 방식이다.

기존에는 파이토치 또는 넘파이로 직접 구현하는 방식 두가지를 배웠었는데, 직관적이고 자세한 내부 로직을 모르더라도 모델학습을 시킬 수 있는 케라스로 훈련하는 방식을 배울 수 있었다.

전처리

그 다음으로 전처리에 대해서는 이론적으로는 알고 있었지만, 학과 수업시간에는 전처리 코드를 어느정도 제공해주셨기 때문에 자세히 알지 못했다.

전처리에 대하여 정말 다양한 방법들이 있고, 데이터를 처리할 수 있는 형태를 잘 잡아놔야 학습이 잘 된다는 사실을 직접 전처리 코드를 구성하며 여러가지를 수정해보니 알 수 있었다.

사이킷 런(scikit-learn)과 판다스(pandas)를 통한 데이터 전처리로 조사하였는데, 특히 판다스는 수업시간에 어떤 교수님이 간간히 언급하던 라이브러리라서 직접 해보니 강력한 라이브러리인 것 같다고 느꼈다.

특히 원핫 인코딩이 중요한 방법으로 알고있었는데, 이론만 알고 있고 코드로는 잘 알지 못하였다가 스터디를 통해 tensorflow 라이브러리와 pandas 라이브러리 두가지 방식으로 구현하는 방식에 대해 배웠다.

주피터노트북

파이썬 수업시간, 인공지능 수업시간에 사용할 때는 몰랐던 유용한 명령어들을 알 수 있었다.

특히 셀 삭제/복제/추가 관련 명령어, 라인 넘버 표시 기능이 유용했다.

CNN

CNN은 합성곱을 통해 구현한 모델로 배워 어느정도 커널, 필터, stride, padding, pooling의 개념은 익히고 있었지만 이것들을 어떻게 설정해야 모델이 학습을 잘하는지에 대해 알 수 있었다.

활성화 함수/최적화 함수(activation function / optimizer)

이 두 함수들의 개념에 대해서도 어느정도 구분지을 수 있었지만, 스터디에서 새롭게 알게 된 것은 활성화 함수에서 어떤 함수가 어떤 분류에 유리한가, 어떤 레이어에서 사용이 주로 되는가에 대하여였다.

이름이 비슷해서 조금 헷갈려서 개념을 다시 정리하자면 출력을 좋은 값이 나오도록 변화시키는 미분함수인 활성화 함수, 출력을 통해 Error값을 알아낼 수 있는데, 이 Error를 줄이기 위해 최적의 가중치값을 찾는 최적화 함수이다.

교차검증(Cross-Validation)

스터디에서는 K-Fold 검증에 대해서 중점적으로 다뤘는데, 데이터 셋을 k로 분할하고 k번 진행하는 방식의 교차검증이고 이는 학과 수업시간에서 배우지는 않았던 내용이었기 때문에 더 정밀한 검증을 위한 방법에 대하여 알 수 있었다.

OX분류

전처리 과정에서 경로에서 o인지 x인지 따라 경로를 설정하는 부분에 대하여는 처음 해보는 부분이었다.

MLP와 CNN 두 가지 방식으로 다 해보았고, 이진 분류에 대하여는 처음 진행했기 때문에 (기존에 해본 것은 MNIST 손글씨 이미지셋 - 0~9 숫자) 이 경우에는 어떤 Optimizer를 사용해야 좋은지에 대해 알 수 있었다.

학습시킬때 Seed 개념에 대해서도 어느정도 이해가 되었다.

인공지능 수업에서는 CNN이랑 Transformer 모델을 다뤘지만 CNN에 대해 잘 이해하지 못했던 것 같은데, 스터디에서 CNN으로 만들어봄으로써 Output 개수 설정, 필터 크기, 패딩 설정을 어떻게 해야 좋은 결과가 나오는지에 대해 알 수 있었다.

그리고 마지막에 Dense층을 여러 개 추가 하는 것이 더 정확도 향상에 도움이 된다는 것도 알게 되었다.

OX분류 개선

개선과정에서 EarlyStop이나 Dropout을 적절히 사용해 정확도를 개선할 수 있었는데, 적은 데이터셋에서 Dropout의 비율 설정이나, 필터크기, 출력층 수, 초기화 방법, 배치정규화를 여러가지 조절해보면서 어떤 식으로 해야 할지에 대해 조금 더 감이 잡혔다.

시각화

시각화 하는 것도 수업때 제공해주신 코드나 인터넷에서 코드를 보고 그냥 붙여넣는 방식으로 하여 방법에 대해 자세히는 잘 몰랐는데, 모델을 이용하여 이진화 시키고 제목이나 출력을 통해 O나 X를 적절히 예측한 값과 실제 정답의 차이를 표기하는 과정을 조금 더 이해할 수 있었다.

Opencv

Opencv는 마우스 입력을 받아 그릴 수 있는 캔버스 또는 함수를 이용해 선을 그려주는 GUI 도구였는데, 스스로 데이터를 Opencv를 통해 그려서 만들고 모델을 직접 어딘가에 적용해서 결과를 적용하는 것에 대해 직접 해보니까 재미있었고, 검증 정확도에 비해 정답을 생각보다 잘 못맞춘다는 점이 조금 아쉬웠다.

마지막으로 모델을 사용할 때에 모델에서 원본 사이즈를 주는 것이 아니라 전처리 시켜서 모델에 들어가는 데이터와 똑같은 형식으로 해줘야 한다는 것에 대해 알 수 있었다.

총정리

인공지능 스터디를 통하여 자세하게 활성화함수, 최적화함수 같은 것들을 정의한 것은 아니었지만 오히려 그 덕분에 직접 모델을 훈련시키고 적용하는데에 있어 난이도가 내려가서 스터디하는 동안 다양한 인공지능 이론들을 코드로 적용시켜볼 수 있어 좋았다.