대본 11p

두번째 발표를 맡게 된 정희석입니다. 제가 발표하게 될 부분은 프로젝트 진행 사항입니다. 크게 진행과정, 데이터 수집 및 분석, 데이터 정규화 및 모델링, 예측 GUI 프로그램 구현의 4가지의 큰 틀을 가지고 진행하겠습니다.

12p

먼저 저희 프로젝트의 진행 과정을 전체적으로 말씀드리면 시뮬레이션 프로그램에서 가상 데이터 수집 및 분석을 진행하고 실제 차량 데이터와 비교, 이를 기반으로 시뮬레이션 데이터 정규화를 진행하였습니다. 다음으로 정규화 방식을 개선하고 예측용 딥러닝 모델을 구현 및 개량하였으며, 마지막으로 예측결과를 한 눈에 알 수 있는 UI 프로그램을 작성하는 흐름으로 진행하였습니다.

13p

예측할 차량 동작 결정에는 흔히 차량의 동작이라고 하면 생각나는 가장 기본적인 차량의 동작인 직진, 정지, 좌/우 커브, 회전, 차선 변경의 총 8가지의 상태로 정했습니다.

14p

저희는 시뮬레이터로써 유로트럭 시뮬레이터 2를 사용하였으며 데이터 수집 장면 사진에서 보시는 것과 같이 팀원 전원이 학습 데이터를 수집하였습니다. 데이터 수집 코스는 프랑스 칼레 지역의 맵을 데이터 수집장소로 정하여 샘플 데이터를 수집하였습니다.

15p

샘플 데이터로부터 시뮬레이터 데이터에서 나오는 데이터 종류와 실제 센서 데이터의 데이터 종류를 비교하여 어떤 센서 데이터가 사용하고 사용할 지를 선별하였고 이 과정에서 총 6종류의 데이터를 사용하기로 하였습니다. 여기서 전후 가속도는 사용하지 않는데 이는 시뮬레이터 데이터의 값이 실제 데이터 값과 차이가 너무 크고, 분포가 안정적이지 못했기 때문입니다. 이후 학습용 데이터를 습득하여 주행 영상을 보면서 차량이 어떤 동작을 하는 지에 대해 라벨링 하는 과정을 팀원 전원이 같이 진행하였습니다.

16p

다음은 선별하고 모은 데이터의 값 분포를 분석해보았습니다. 화면에 보시는 것과 같이 각 센서 데이터의 범위를 확인해보고 학습용 데이터와 실제 데이터의 데이터 분포를 분석하여 5%~95% 사이의 값을 기준으로 정규화를 진행하였습니다.

17p

이후 정규화 과정에서는 앞의 데이터 분포에 따라서 시뮬레이터 데이터를 실제 데이터와 유사하게 만드는 1차 정규화, 범위제한을 통해 예외 값을 처리, 2차 정규화를 통해 실제 데이터의 분포에 맞추는 과정을 거치게 됩니다.

18p

초기 모델링은 학과에서 배운 수업시간에 사용했던 예시 모델을 통해 진행해 보았습니다. 단순히 Tensor Layer를 나열한 DNN 모델과 Convolution Layer와 MaxPooling Layer를 여러 개 통과하는 CNN모델을 각각 구현 및 학습을 시킨 결과 각 동작에 대해 50% 미만의 정확도를 확인할 수 있었습니다.

19p

이후 그나마 정확도가 나은 CNN모델을 사용하기로 결정하고 Fully Connected Layers를 통과하기 직전의 Layer에 사용 가능한 방식이 Flatten과 GAP, Global Average Pooling방식이 있는 것을 확인하고 이를 사용해 보았습니다.

20p

다음은 저희가 설계해 본 CNN 모델입니다. 두 모델은 GAP, Flatten의 여부만 다르고 여러 Convolution Layer를 공통적으로 통과하는 모델입니다. 다음 층으로 넘어갈 때 각 Filter의 최대값을 선택하는 MaxPooling Layer 대신 Stride 옵션을 통해 예측용 데이터의 차수를 점차적으로 줄이는 식으로 구현해 보았습니다. 여기에서 Window\_Size를 48로 설정하였는데 이는 데이터가 1초당 10개씩 나오며, 한 동작이 일어나는 시간이 대략 4.8초로 예상되었고 실제로 Window\_Size를 다양하게 해서 테스트했을 때 48이 가장 좋은 결과를 보여주었습니다. 그리고 두 CNN 모델의 예측 결과를 조건문을 통해 직진, 좌/우 차선, 정지의 예측결과는 GAP Layer를 사용한 CNN 모델에서, 좌/우 커브, 회전의 예측결과는 Flatten Layer를 사용한 CNN 모델에서 선택하도록 구현 하였습니다.

21p

마지막으로 데이터 분석 프로그램입니다. 데이터 분석 프로그램에 사용된 모듈은 UI를 표현하기 위한 PyQt5, 주행 영상 재생을 위한 Opencv Python, 학습된 모델을 불러와서 예측을 진행하는 TensorFlow, Keras 모듈이 사용되었습니다.

22p

왼쪽 그림에 보이는 학습된 예측용 모델이 h5 확장자를 가지는 파일로 UI프로그램에서 load되며 csv파일과 영상 파일의 이름이 일치하게 되면 예측이 진행되도록 구현하였습니다.