김신영 포트폴리오

프로젝트

1. 2022년 여름방학 데이터청년캠퍼스 Sequence to Sequence Learning with Neural Networks 논문 리뷰

목표 : 코드 기반으로 논문의 이해, 정방향과 역방향의 결과 분석

접근 과정

발표 내용

관련 코드 1, 관련 코드 2

느낀점

- 처음해본 논문 리뷰

논문과 다른 사람들의 논문리뷰를 통해서 논문을 이해하고 발표하는 과제였다.

팀원이 7명이었기에 각자 관심있는 분야를 나눠서 발표를 진행하였다.

코드 기반으로 발표를 진행하다보니 코드를 구현한다는 것이 **논문의 내용을 확인하는 정도에 불과**하다는 것과

코드를 설명할 수는 없기 때문에 **보여줄 수 있는 것이 많지 않다**는 것을 느낄 수 있었다.

2. 2022년 여름방학 데이터청년캠퍼스 팀프로젝트 : yolo기반 데이터 증강, 준지도학습을 활용하여 화상 이미지 분석

목표 : 화상 이미지 데이터를 통해 화상의 정도 판별

접근 과정

최종 모델을 기반으로 CSA 2022 The 14th International Conference on Computer Science and its Applications 학회에 논문 제출 : 한글, 영문

관련 코드 1, 관련 코드 2

느낀점

- 처음해본 팀프로젝트와 논문 작성

다른 팀원 한분과 함께 객체인식 모델을 맡게 되었다.

화상이미지는 의료데이터로 구하기가 쉽지 않기 때문에 **적은 양의 이미지 데이터**를 **교차 검증**, **이미지 증강**, **준지도학습**을 통하여 정확도를 높이고자 하였다.

yolov5과 albumentation 라이브러리를 통해 활용하고자하는 이미지 증강 기법과 객체인식 모델의 형식을 맞춰서 구현하였다.

최종 모델은 초기 모델과 비교하였을 때 F1 Score기준으로 0.38에서 0.575로 51.32% 향상되었다.

3. 2023 1학기 캡스톤 프로젝트

목표 : stable diffusion을 활용하여 클래식 음악 데이터를 통해서 이미지의 분위기 변환

관련 코드

느낀점

활용한 데이터가 음원으로 매우 큰 데이터였다.

주요 실행 환경으로 Google Colab Pro를 사용하였는데 제한된 컴퓨터 성능으로 인해 런타임이 빈번하게 나가는 것이 문제였다..

따라서 **데이터의 차원을 MFCC방법을 통해 축소**하여 데이터의 크기를 줄였고

각각의 분위기에 대해 **One vs Rest의 분할 모델**로 접근하여 한번에 학습시키는 모델의 크기를 줄였다.

4. 2023 여름방학 SW중심대학공동AI경진대회

목표 : 위성 이미지 데이터를 통해 건물 영역 분할

관련 코드

느낀점

단일 모델만 접근했을 때 segformer모델이 Unet모델보다 더 향상된 결과가 나왔다.

하지만 **segformer 모델에 여러 이미지 증강 기법을 적용하였을 때 Unet에 여러 이미지 증강 기법을 적용한 것보다 좋지 못했다.**

무작정 증강, 변수 처리를 적용한다고 해서 항상 가장 좋은 결과가 나올 것이라는 것을 장담할 수 없다는 것을 느낄 수 있었다.

이때부터 모델 구조는 블랙박스로 결과만 보고 판단하여야한다고 배웠지만 어떻게 모델이 판별해가는지 모델 구조를 살펴볼 수 있는 방법에 대해서 고민하게 되었다.

5. 2023년 2학기 딥러닝 수업 팀프로젝트 : Child Mind Institute - Detect Sleep States

목표 : wrist-worn accelerometer data를 활용하여 수면 상태의 감지

관련 코드, 최종 레포트, 최종 발표

느낀점

제한된 컴퓨터 성능이 제공되는 캐글 환경에서 프로젝트를 진행하였다.

MultiResidualBiGRU모델과 Deberta모델의 데이터 전처리 부분을 맡아서 진행하였다.

가장 많은 성능의 향상을 이끌어낸 **변수의 차분**은 맨 처음에 변화량에 중점을 두고자 하는 의도로 시도되었다.

프로젝트를 진행하면서 **off-line 상태**의 경우 진동 등으로 인해 **반복되는 값이 존재**하다는 것을 알 수 있었고 차분을 통해 해당 상태의 값을 0으로 만들 수 있어 성능의 향상을 가져왔다고 판단할 수 있었다.

또한 가속도 변수인 enmo의 처리에서 0에 가까운 값을 크게 처리하는 것이 Bi-GRU모델에서는 성능의 향상을 가져왔으나 Deberta 모델에서는 심각한 성능의 감소를 가져왔다.

모델 자체에서 양방향 학습이 이뤄지기 때문에 데이터의 방향을 역전시켜 활용한 증강은 의미가 없었다.

다른 사람이 올린 Discussion에서 데이터를 간단한 plot으로 그린 후에 객체 인식을 통해서 처리하여 높은 성능을 가져온 방법이 인상깊었다.

6. [중단] 2023년 2학기 sw연구프로젝트 및 실습 수업 프로젝트 : 강화학습 환경 구축에서 종료

목표 : 2차원 데이터를 3차원으로 변환 후 임의의 평면을 짤라 2차원 데이터로 변환

관련 코드

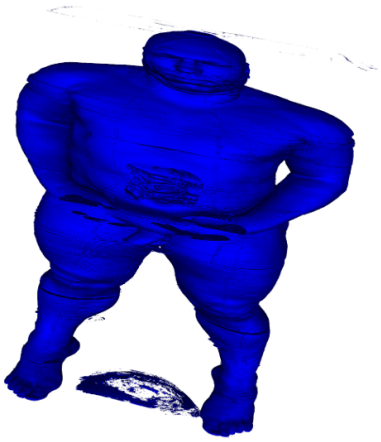
느낀 점

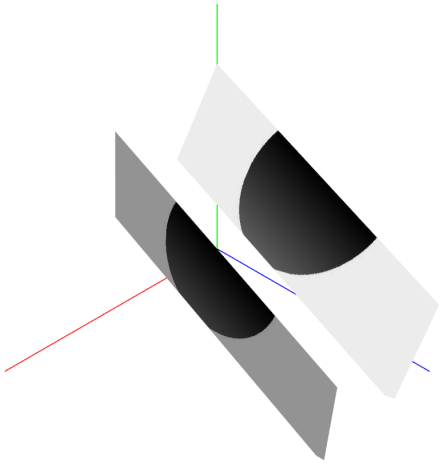
vtk 소프트웨어를 활용하여 팀원이 cleaning한 2차원 데이터를 쌓아 3차원 데이터로 만든 후 mesh 알고리즘으로 처리하였다.

이후 벡터 2개를 통해 평면을 정의하였고 3차원 데이터를 평면의 법선벡터로 짤랐다.

하지만 여기서 기울어진 평면의 경우 해당 평면을 2차원 데이터로 변환하지 못하고 학기가 끝났다.

하지만 이후 ray tracing이라는 키워드를 알게 되었고 이를 통해 사영기하학이라는 개념을 통해 2차원 데이터를 변환하였다.





7. [진행중] 2023년 스타크래프트2 build 강화학습

목표 : 스타크래프트2 빌드를 강화학습을 통해서 최적화

관련 코드

느낀점

1. 스타크래프트2의 라이브러리를 활용하여 먼저 if문으로 강화학습으로 들어갈 명령을 구현하였다.

이 과정 속에서 ASSIMILATOR에 들어간 workers의 정보가 누락된다는 것과 자원 최적화하는 distribute_workers 함수의 문제점을 찾을 수 있었다.

운영체제 수업에서 배운 여러 lock 개념을 통해서 시도하며 가스에 일꾼을 이동시키는 것과 가장 가까운 일꾼이 넥서스 인접 지역에 가스를 건설하게 하였다.

2. 특정 빌드의 최적화를 목표로 spinlock처럼 매 프레임마다 상태, 액션을 처리하는 환경을 구성하여 강화학습을 하고자 하였다.

하지만 매 프레임마다 액션을 계산할 때 실행되지 않는 명령이 더 많아 제대로 학습이 되었는 지 평가하기 어려워 현재 액션이 끝나고 난 후에 다음 액션을 받는 형태로 처리하였다.

모델 학습에서 cpu가 100%라.. 지인의 노트북 환경에서 학습을 진행하였다..

복잡한 상태를 분석하는 것이 어려워 먼저 쉬운 문제부터 해결해보고자 하였다.

3. grid search보다 더 효율적으로 최적화된 빌드를 찾아야 하기에 back-tracking으로 가능한 모든 경우를 생성하여 기록을 저장하였다.

질럿 5기의 모든 경우 : 1014784가지, 질럿 생산 이후 일꾼 생산 x, 현재 생산 위치 기준 파일런 필요한 경우에 짓는 경우 : 4801가지

4. 이후 노트북 2대를 활용하여 간단한 socket 통신을 통해서 강화학습 모델과 강화학습 환경을 따로 돌리게 하였다.

5. 연속적으로 모델 학습 가능한 환경을 구축하였다.

이 과정 속에서 불가능한 경우를 환경에 대한 액션을 줄 때 계산하여 처리하게 하였다.

불가능한 액션을 선택할 때 대부분 현재 위치 그대로 오게 하여 액션 수를 늘려 감가율을 통해 더 짧은 액션을 주게 한다.

하지만 여기에서는 액션이 긴 경우가 더 빨리 끝나는 경우도 존재하여 감가율을 사용하지 못한다.

6. 강화학습 책을 토대로 라이브러리를 가져와 맞추는 것이 아닌 필요한 기능을 가진 코드를 구현하였다.

프레임이 빠를 수록 더 큰 보상을 주기 위해 지수함수를 최종 보상으로 주었다.

최종 완료 시점을 기준으로 중간 명령을 프레임의 비율로 보상을 줄 경우를 주고자 하였다.

이 경우 일꾼을 생성하는 경우는 바로 실행되어 0,1 프레임에 계산된다. 1을 일괄적으로 가산하여 처리할 수 있다.

하지만 더 긴 최종 프레임에서 중간 과정의 보상이 더 큰 경우가 존재하여 해당 방법은 제대로 학습되지 않는다.

7. 질럿 2기를 생성하는 경우를 학습할 때 상태와 액션에 대한 가치를 분석하기 위해 시각화하였다.

질럿 2기의 모든 경우 : 318가지, 질럿 생산 이후 일꾼 생산 x, 현재 생산 위치 기준 파일런 필요한 경우에 짓는 경우 : 177가지

잘못된 액션의 경우 바로 끝나기 때문에 -보상을 주지 않으면 목표까지 잘 찾아가지 못한다.

하지만 -보상의 영향을 바로 직전 상태에만 주지 않고 이전 찾아간 상태에 주는 경우 이전 지식을 바탕으로 환경을 제한한 영향을 크게 받는다.

환경의 끄트머리에 있을수록 더 빠른 경로가 있음에도 불구하고 해당 경로로 잘 가지 않는다.

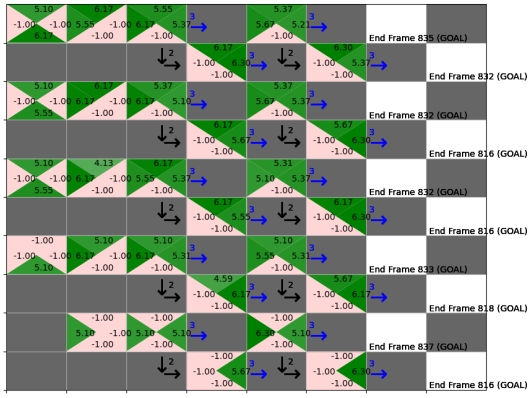
8. imitation learning

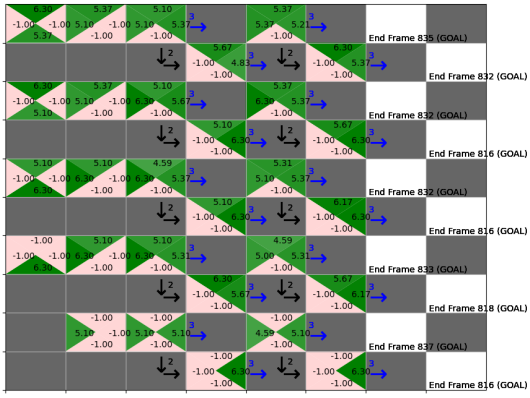
이미 알고 있는 정보를 기반으로 초기 경로를 1 or 2번 주었을 경우 해당 경로를 중심으로 가지치기 하듯이 퍼져나간다.

정확한 정답을 줬을 경우와 근사한 경우, 잘못된 경우를 줬을 때 영향을 분석 중...

강화학습 환경 체크1 강화학습 환경 체크2 socket 통신 환경 체크 PPO 학습

학습 결과 시각화 예시





8. [진행중] 2024년 mycobot320을 이용한 imitation learning

목표 : mycobot320을 imitation learning으로 동작

관련코드

진행 과정

2024.7.31 : aloha코드를 기반으로 예시를 돌릴 수 있게 환경 설정 및 오타(?) 해결

2024.8.02 : aloha코드에서 imitate_episode.py 이해

2024.8.04 : aloha코드에서 policy.py 이해

2024.8.13 : sim_move_cube_scripted라는 에피소드 설정

2024.8.14 : sim_move_cube_scripted 에피소드를 위해 xml 파일 설정, 돌려보고 관련 내용로 정리

2024.8.16 : sim_move_cube_scripted 에피소드를 one arm으로 설정(모델 입출력 수정)

2024.8.30 : sim_move_cube_scripted 에피소드를 one arm으로 50개의 데이터를 100000번 학습 시켰으나 일정 수준 이상 성공률이 올라가지 않음

[카메라를 2개로 하면 학습 시간이 카메라 1개에 비해 10배 늘어남, 더 작은 epoch로 성공하는 시점이 발생하나 시간상으로는 더 느림]

2024.9.07 : mycobot320으로 sim_move_cube_scripted 에피소드 구현

2024.9.12 : mycobot320으로 10개의 데이터를 5000번 학습 후 결과 확인

2024.9.20 : 카메라가 목표를 보고 있는 상황에서 시도, chunk를 1초, 2초 단위의 연산량으로 잡고 처리

[chunk가 작을 수록 모델 학습 및 추론 시간 감소, ~~해당 DT(1/FPS)만에 갈 수 없는 위치를 계산하는 경우 존재~~]

[고정 카메라에 그리퍼가 보이지 않아 물체를 잡아야하는 순간과 잡고 있는 순간을 구분 못하는 듯?]

2024.9.23 : mycobot320 조작하는 코드도 라이브러리에 추가, 폴더 정리

2024.9.25 : 에피소드를 더 잘게 쪼개서 기능별 시도, 하드웨어 문제 파악 및 해결! 성공!! 만세!

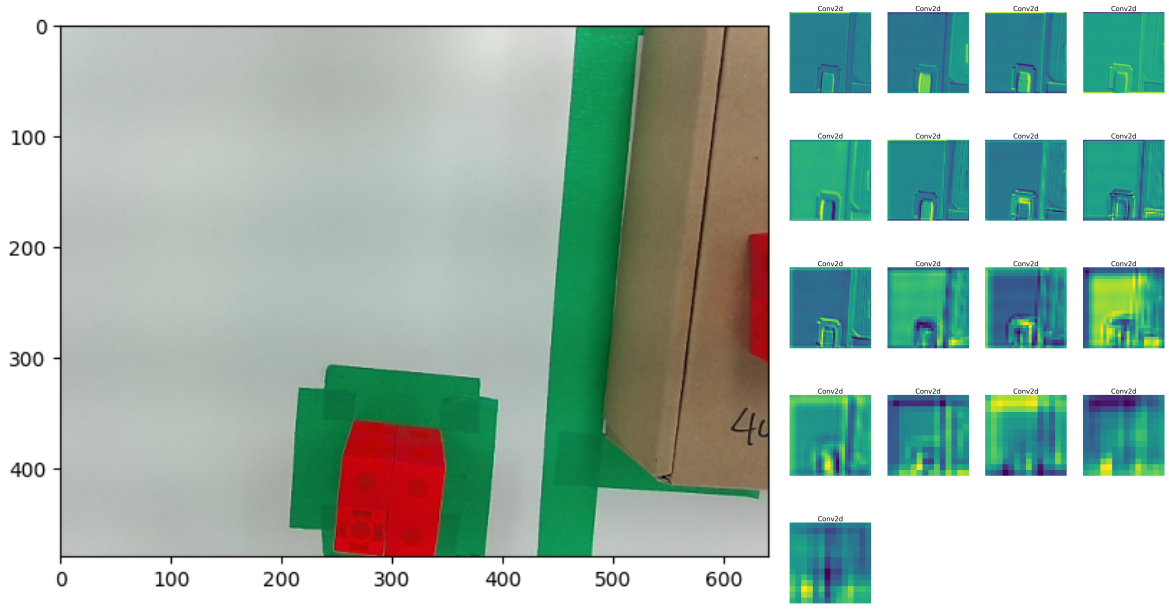
진행 과정

2024.9.27 : 상황별로 다른 행동을 하도록 에피소드 구현

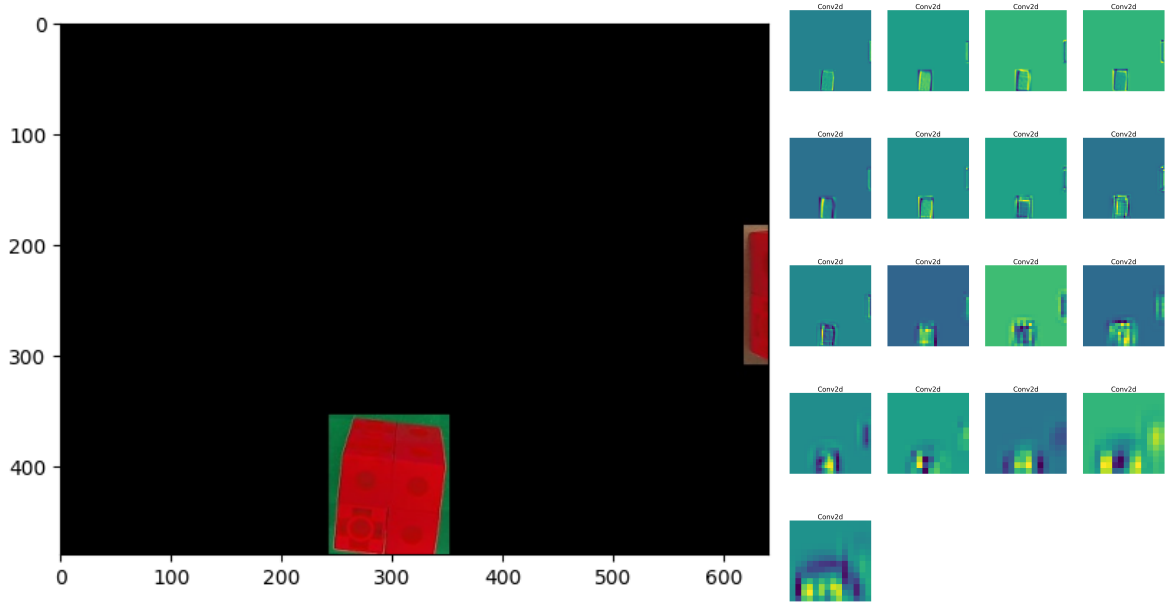
2024.9.30 : 확률적으로 동작하거나, 한 동작만 하는 문제 발생

[chunk가 클수록 현재 상태에서 다음 상태를 더 고려하기에 행동이 부드러워지지만 좀 더 줄여야 할 듯?]

2024.10.31 : 상황에 따른 판단 분석을 위해 resnet18의 feature map을 살펴봄, imitation의 특성에 따라 간단한 masking하는 코드 생성, chunk size 75로 결정



masking



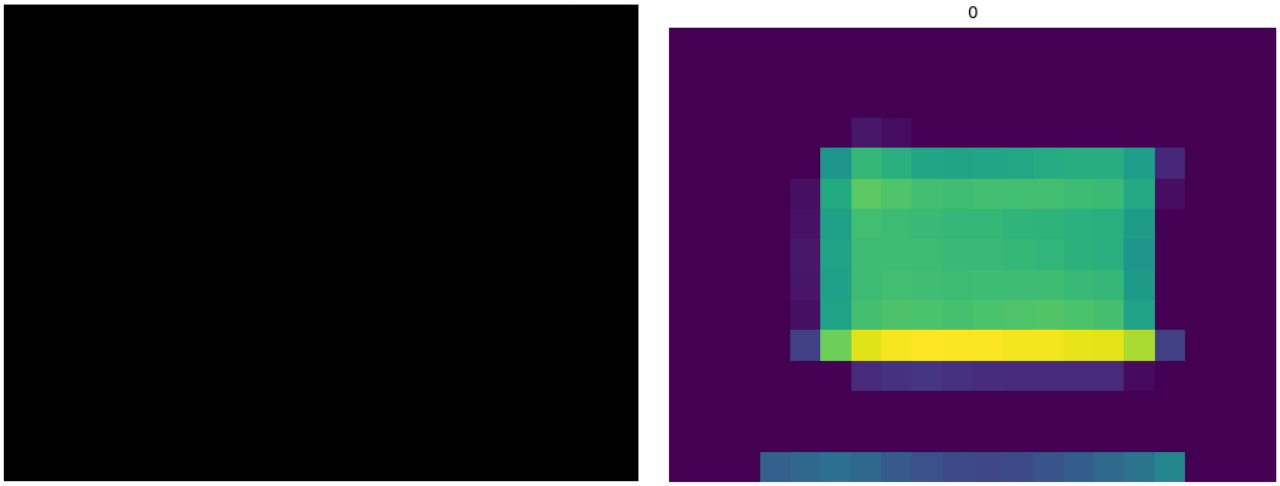
[imitation의 특성에 따라 배경을 분류해낼 수 있고 카메라의 노이즈는 회전변환행렬을 통해서 매칭 시킬 수 있을 것임]

[하지만 mycobot320의 로봇암과 연결된 카메라 선의 위치가 매번 바껴서 배경 제거할 때 고려해야 함]

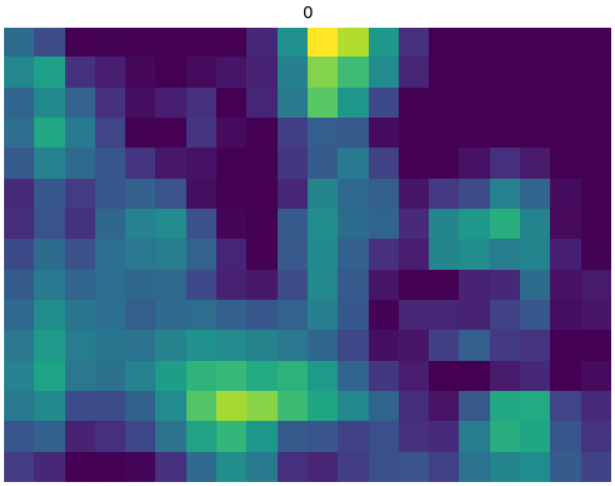
[하지만 주요 박스를 제외하고 모두 마스킹하면 공간적 정보를 인식하지 못해서 제대로 행동하지 못하는 듯!]

2024.11.14 : 이전 실패한 모델의 feature map 분석, 데이터 전처리를 통해 2가지 행동 중 무엇을 해야할 지 판단하여 동작 성공!!

첫번째 행동으로 동작한 2번째 카메라의 값이 모두 0일 때의 이미지와 feature map



첫번째 행동으로 동작해야 하지만 두번째 행동을 할 때의 2번째 카메라의 이미지와 feature map



하나의 모델로 2가지의 행동을 판단하여 동작

2024.11.19 : Residual Attention Network for Image Classification논문과 구현 링크를 참고하여 backbone의 차원을 맞춰 교체 시도, 모델의 경량화 필요성을 확인, Optimizer를 SGD, Adam로 설정하는 것에 따른 성능 차이를 확인

[resent151x2 혹은 resnet50, residualattentionmodel92는 모델 크기가 180M 넘어가서 gpu 메모리 한계가 걸림.. resnet18 기준으로 새로 모델을 짜서 경량화 시도 중]

2024.11.21 : 구현 링크를 참고하여 ResidualAttentionModel56U의 성능 테스트, resnet18과 ResidualAttentionModel을 바탕으로 새롭게 작성한 모델(Resent18AttentionModelU) 성능 테스트, CIFAR-10 데이터셋으로 평가

[ResidualAttentionModel56U : Total params : 7,202,794, Estimated Total Size (MB) : 107.90, Number of model parameters : 11413482, epoch 100 Test Accuracy : 93.86%]

[Resent18AttentionModelU : Total params : 459.114, Estimated Total Size (MB) : 7.59, Number of model parameters : 723306, epoch 100 Test Accuracy : 88.01%]

알고리즘

코드트리

2022년 여름방학 학교지원으로 코드트리 시작

2022년 겨울방학 코드트리 코딩테스트 대비 캠프

2023년 여름방학 코드트리 코딩테스트 대비 캠프

코드트리 깃허브 연동 전 블로그에 작성, 코드트리 깃허브 연동

그누빌

2023년 1학기 알고리즘 스터디 : 취업과 이직을 위한 프로그래머스 코딩 테스트 문제 풀이 전략 : 파이썬 편

2023년 2학기 알고리즘 스터디 : Do it! 알고리즘 코딩 테스트: 파이썬 편

학교 활동

2022년 1학기 우리 함께, 코딩해봄 C 튜티 : 수업 내용

2023년 2학기 코딩존 조교

기억하고 싶은 문제

1. 2022년 1학기 자료구조 수업 : 아군적군

목표 : Union-find 자료구조를 활용하여 각 번호에 할당된 사람의 아군, 적군 정보 관리

접근 과정, 풀이

느낀점

처음에 풀고나서 도저히 어디서 틀린지 모르겠어서 수업때 교수님께 테스트 케이스를 하나만 더 열어달라고 하였다.

하지만 그 때 해당 테스트 케이스를 다 뜯어본 결과 오류가 있다는 것을 알 수 있었다.

그렇게 1차 문의를 드린 후에 코드가 개정되었다.

하지만 그 이후에도 문제가 없는데 틀려서 이번에는 가능한 **모든 경우의 테스트 케이스를 만들어서 검증**해보고 다시 조교님께 메일을 보냈다.

그렇게 2차 문의를 드린 후에 코드가 개정되었고 처음에 짠 코드가 맞다는 것을 알 수 있었다.

이런 경험을 통해서 코드를 구현하고 맞았으면 끝이 아니라 코드가 옳게 돌아가는지 체크해보며 자신이 짠 코드에 확신을 가질 수 있도록 **증명**할 수 있어야한다는 것을 느낄 수 있었다.

2. 2023년 1학기 고급문제해결기법및실습 수업 : 계란문제

목표 : DP에서 시간복잡도를 더 줄이기 위해서 DP 테이블을 수열로 보고 일반항을 계산

접근 과정

느낀점

교수님이 시간복잡도를 더 줄일 수 있다고 하셔서 어떻게 해야 더 줄일 수 있을 지 고민하다가

DP 테이블의 값을 수열로 보고 일반항을 계산하고자 하였다.

처음에는 **마르코프 연쇄 법칙**으로 인해서 반드시 계산될 것이라고 생각하고 접근하였다.

조합을 통해서 일반항을 계산해낼 수 있었지만 시그마가 들어감에 따라서 시간복잡도를 더 줄일 수는 없다는 결론에 도달하였다.

하지만 **DP 방법이 아닌 이진탐색과 조합**을 통해서 풀 수 있다는 것을 알 수 있었다.

또한 이 과정을 통해서 마르코프 연쇄 법칙에 대해서 잘못 인지하고 있었던 점을 알 수 있었고

조합을 계산하는 여러가지 방법에 대해서 찾아볼 수 있었다.

3. 2023년 1학기 설계패턴 수업의 프로젝트 : 계란문제의 여러 풀이를 학습할 수 있는 프로젝트를 디자인 패턴을 적용하여 작성

목표 : 여러 풀이 방법에 대한 설명과 디자인 패턴의 적용

관련 코드

느낀점

디자인 패턴을 배울 때는 예시로 배워 이해할 수 있었지만

디자인 패턴을 적용하는 양이 많아질수록 처리하는 과정이 쉽지 않다는 것을 느낄 수 있었다.

계란문제에 대한 여러 풀이 방법을 정리하면서 DP테이블에서 for문으로 접근하는 코드에서 이진 탐색으로 처리할 수 있는 등

내가 놓쳤던 점과 이진탐색과 조합을 이용하는 방법에서 더 개선된 방법을 찾아낼 수 있었다.