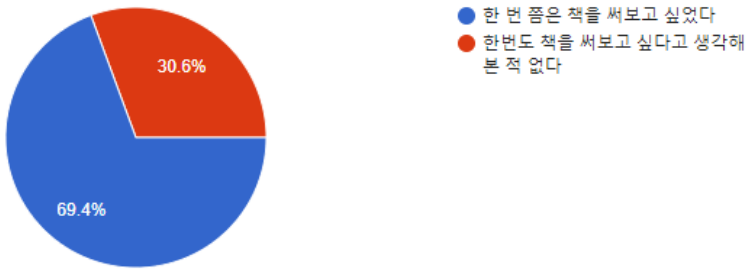
최종 프로젝트 랩업 리포트

프로젝트 개요

프로젝트 주제

> GPT-3 기반 텍스트 생성 + CycleGAN을 이용한 동화풍 이미지 변환을 합친 `나만의 동화 만들기` 서비스

- Why?
  - 코로나 19 이후 디지털 기반 콘텐츠 소비의 확대로 특히 전차잭 시장의 성장과 나만의 전자책 만들기 콘텐츠가 인기를 끌음
  - 디지털 콘텐츠 시장의 성장에 따라 웹 기반 서비스로 가닥을 잡음
  - 아이부터 어른까지 전 연령대가 접근할 수 있는 장르적 특성에 따라 `동화`를 주요 아이템으로 선정
- 프로젝트 배경
  - 창작의 주요 걸림돌은 `부담감`, `아이디어`, `시간`이라는 설문 조사 결과를 얻었다.
  - 특히, 한번이라도 글쓰기를 해보고 싶다는 비율이 **69.4%**인 만큼, 글쓰기를 돕는 서비스가 있다면 많은 이용을 기대할 수 있을 것이라 생각
  - AI 기술을 활용하면 창작의 걸림돌이 되는 부분에 대해 사람들이 도움을 얻을 수 있을 것이라고 판단
  - 따라서 GPT 계열의 텍스트 생성 모델을 이용해 창작을 하려는 사람들에게 일종의 `콘티`를 제공해주는 프로젝트로 결정



프로젝트 아키텍쳐

- 프론트엔드 & 백엔드
  - React.JS, FastAPI 기반의 웹 서비스
- 모델
  - 동화 생성 모델
    - kakaobrain/KoGPT, SKT/KoGPT2와 SKT/KoGPT-trinity를 시험 및 비교 후 `KoGPT-trinity`로 결정
    - kakaobrain/KoGPT의 경우 GPT-3 기반으로 가장 좋은 성능을 보였으나, parameter 숫자가 6.2B로 너무 큰 모델이여서 주어진 환경에서는 학습 및 추론이 **불가능** 하였음.

        - KoGPT-trinity는 GPT-3 기반의 한국어 모델로, KoGPT-2 대비 (**125M → 1.2B**) 훨씬 많은 parameter 숫자를 가져 **훨씬 좋은 생성 능력**을 보임.
        - GPT-3 기반임에도 비교적 적은 parameter 숫자를 지녀서 준비된 개발환경에서도 사용 가능하였고, 속도도 준수하였음
        - 생성 모델인 만큼 정량적인 평가가 불가능하여 보조지표로 **Perplexity**를 이용하였음
    - 이미지 모델
        - 프로젝트 초기 단계에서는 **VGGNet**을 이용한 Style transfer를 동화풍 이미지로 변경
        - 이후 `CycleGAN`을 사용하여 동화풍 이미지로 변경하도록 모델을 학습시켜 사용함
        - CycleGAN을 학습시킨 데이터는 동화풍의 이미지 데이터와 일반 이미지를 Style transfer를 사용하여 동화풍으로 변경한 이미지 데이터도 추가로 사용함

데이터

- 동화 데이터 **(4.78MB)**
    - 청와대 어린이 전래동화 100건
    - 국립국어원 모두의 말뭉치 - 비출판물 데이터
    - 그림형제 동화 번역본
    - 이솝우화 번역본
    - 2021 ~ 2022 신춘문예 동화 당선작 - 강원일보 외 15개 신문사
- 그림 데이터
    - pixabay 등 저작권 free 사이트에서 동화풍 이미지 150장 수집
    - CycleGAN 공식 학습 이미지 apple2orange, summer2winter, horse2zebra 등 논문에 기재된 데이터셋 3.2GB
    - VGG 모델을 활용한 Style image

프로젝트 팀 구성 및 역할

> 자, 연어 한 접시는 **5명**의 팀원으로 구성되어 있습니다.

- 강진희 : 데이터 수집 / 키워드 추출 / 이미지 검색 API / 문서화
- 김태훈 : 데이터 수집 / 데이터 전처리 / 서비스 아키텍쳐 구성 / UI UX 디자인
- 이도훈 : 데이터 수집 / 데이터 전처리 / 문장 생성 모델 / 프론트엔드 구축
- 차경민 : 데이터 수집 / 데이터 전처리 / Style Transfer / StyleGAN
- 김선재 : 데이터 수집 / 문장 추천 모델 / FastAPI / DB 구축

프로젝트 수행 절차 및 방법

🕑 프로젝트 타임라인

2022.05.09 ~ 2022.06.10

1주차 아이디어 탐색

- 주제 `탐색 및 아이디어` 도출 단계
- 한계를 고려하지 않고 자유로운 의견을 내어 종합함

2주차 아이디어 구체화 및 문서화

- 현실적 한계 (시간, HW, 이론적 한계)를 고려하여 아이디어를 선별
- 아이디어 구체화 및 문서화를 통해 서로간의 이해가 다른 부분을 확인하고 아이디어를 다듬어 나감
- 프로젝트 `기획서` 및 `간트 차트`를 작성하여 향후 프로젝트 진행 계획 확립

3주차 모델 탐색 및 파인튜닝

- 프로젝트 기능 구현에 필요한 `모델들을 탐색`하는 과정을 가짐
  - 우리가 원하는 Task를 다양한 모델들로 `실험 및 비교`
  - 실시간 웹 서비스인 만큼 가장 중요한 요소는 `시간`으로 기준을 잡고 진행
  - 이후 확정된 모델을 요구조건에 맞춰 `파인튜닝`을 진행
- 그 외 맞춤법 검사 API, 키워드 추출, 감성 분석 등 요구되는 다른 기능의 구현도 진행

4주차 프로젝트 프로토타이핑 및 테스트

- 프로토타이핑을 위해 3주차 결과에서 모델을 확정함
  - 텍스트 생성은 `KoGPT-trinity`, 이미지 변환은 `CycleGAN`으로 확정
  - 프로토타이핑을 위해 inference 파이프라인 구축
- 파인튜닝을 마친 모델을 직접 inference 하는 과정은 불편함
  - 다양한 사람들에게 사용을 해보고 의견을 수집하기로 결정하여 **프로토타입** 제작 작업을 시작함
  - **Streamlit**을 이용하여 빠르게 프로토타이핑을 마쳤고, 이후 `서울 국제도서전` 및 `부스트캠프` 및 주변 사람들을 통해 필드 테스트를 거침
- 설문 조사 결과를 바탕으로 문제점 추가발견 및 각종 이슈를 확인하고 개선하는 작업을 진행함
  - 너무 많은 옵션이 존재: `문장 길이` 및 `temperatrue`와 `repetition penalty`만을 남기고 다른 옵션은 숨김
  - 문장의 어미가 자주 바뀜: 학습 데이터의 문제라고 판단, 데이터셋 구성을 전체 corpus가 아니라 `동화별` 세트로 나눠 문체를 가급적 통일하도록 유도
  - 특정 단어 입력시 출력이 고정되는 문제: 일부 데이터셋에 편향되어 발생한 문제로 확인, `데이터셋 추가 확보`를 통해 해결
  - 문장 단위로 나오지 않고 중간에 잘리는 것이 불편: 데이터를 문장 단위로 나누고 `스페셜 토큰`을 부착하여 문장 단위로 후처리를 할 수 있게 개선

5주차 프로젝트 서빙 및 모델 개선 작업

- 프로젝트 서빙을 위한 백엔드 및 프론트엔드 작업 시작
  - 프론트엔드: React를 이용하여 구축. heroku를 사용해 배포함
  - 백엔드: FastAPI를 이용하여 백엔드 서버 구축 및 모델 Inference 파이프라인 구축. MongoDB 구축
- 앞서 조사된 여러 이슈와 추가적으로 발견한 이슈의 개선 작업 진행

- 단어 1개만 입력하는 경우 발생하는 문제 : 토큰 1개만 입력으로 들어온 경우 발생하는 문제로 판단함. 사용자에게 최소 2단어 이상 입력하게 하는 방식으로 해결
- 특정 문장이 과도하게 반복되는 문제: `repetition penalty` 및 `top-p, top-k` sampling 수치를 테스트해보며 최적의 값을 찾음

- Perplexity 수치를 살펴보며 텍스트 생성 모델의 성능을 대략적으로 분석함
  - 짧은 문장에서는 큰 차이가 없지만, 긴 문장에서는 큰 차이를 보임

input	KoGPT2	KoGPT2 + 데이터 전처리	KoGPT2 + 동화별 학습	KoGPT-trinity + 동화별 학습
맑은 하늘	5.20	9.08	7.06	2.44
옛날에 고양이 한 마리가 있었어요	6.26	13.01	12.04	1.97
같은 날, 같은 도시에서 태어난 두 아이. 한 아이는 불쌍하게 거지 집에서 태어난 경민이였고 한 아이는 궁전 왕자로 태어난 도훈이였지요	8.01	224.09	177.61	4.59

프로젝트 결과

동화 생성 모델

- GPT-3 기반의 한국어 생성 모델인 skt/KoGPT-trinity를 사용
  - 생성 모델의 대표인 GPT를 사용하기로 하였으나, 기존 GPT의 경우 한국어 생성능력이 약함. 따라서 방대한 양의 한국어 corpus*(한국어 위키 백과, 뉴스, 청와대 국민청원 등)*로 학습된 `KoGPT`를 이용하기로 결정
  - KoGPT의 경우 다양한 한국어 corpus로 학습이 되어있어서 문장 생성 시 기사나 위키피디아의 문체와 같은 글을 생성해냄. 동화에 어울리는 문장을 생성하지 못 함.
  - 따라서 `동화 텍스트 데이터`를 수집하여 파인 튜닝하면 한국어 `동화 생성`에 특화된 GPT 모델을 만들 수 있을 것이라고 예상하고 학습을 진행함
    - 텍스트 데이터는 청와대 전래동화, 그림형제 & 이솝우화 번역본, 신춘문예 당선작 등 다양한 소스에서 수집하여 약 **4MB**의 텍스트 데이터를 학습에 사용함
    - 실제로 파인튜닝 전 후를 비교해보면 훨씬 더 자연스럽게 **동화풍의 텍스트를 생성**하는 모습을 볼 수 있었음
  - 파인튜닝을 마친 모델의 편리한 공유를 위해 **Huggingface hub**에 올려 팀원간 공유 및 사용을 쉽게 하도록 하였음. [링크]

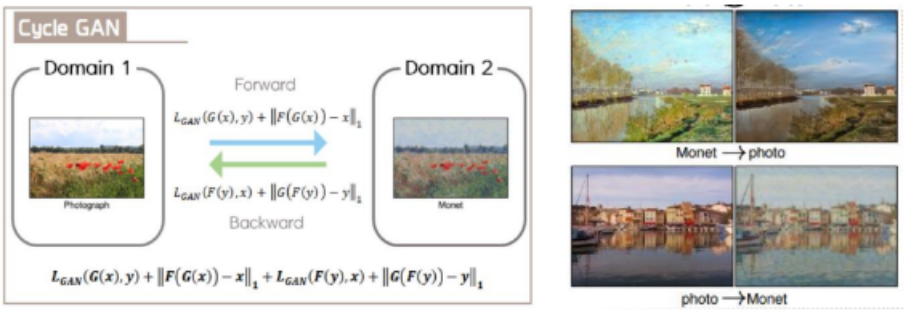
- 데이터 전처리
  - 우선 데이터 상의 불필요한 부분을 수작업으로 제거함. 작가의 주석이나, url 링크, 이모티콘 등을 제거하여 순수한 텍스트만을 남김
  - 처음에는 모든 데이터셋을 바로 사용함. 그러나 이 경우 한 이야기의 끝에서 바로 다음 이야기의 시작이 연결되기 때문에 모델이 문맥을 잘 파악하지 못 하는 현상이 발생함
    - 이를 개선하기 위해 우선 데이터를 스페셜 토큰을 사용하여 문장 단위로 나눔
  - 그럼에도 문장 생성이 길어질수록 전혀 연관없는 이야기가 이어지는 현상이 발생함

- 해당 현상이 전체 데이터를 하나의 파일로 학습하기 때문에, 연관성이 떨어지는 다른 이야기의 문장으로 이어진다고 판단함
- 해당 문제를 해결하기 위해 데이터를 동화별로 나누고 문서단위 batch를 적용하여 문서 단위로 학습하도록 함
- 학습 결과 앞선 모델보다 훨씬 문맥에 맞는, 이야기 구조가 이어지는 문장을 생성하게 됨

```
*Before
[INPUT]: 산신령과 나무꾼이
[OUTPUT]: 산신령과 나무꾼이 오늘 아침 산책을 나가려는데, 두 짐승이 수풀에서 튀어나와 서로 다정한 눈빛을 교환하며 이렇게 말하는 거예요.
 우리 잠시 여기서 눈 좀 붙이고 가지. 내가 지금 너무 피곤해서 서 있을 수가 없구려.
 그래서 산신령이 말했어요.
 네 놈들 뭐 이런다고? 그냥 가서 꿀이나 따 오거라.
*After
[INPUT]: 어두운 도시
[OUTPUT]: 어두운 도시 풍경을 그대로 담아내었다.
 회색 건물에 드문드문 박힌 유리 창문. 눈앞을 아른아른 거리는 백색의 그림자와 진주 같았다.
 도시에 비친 해는 붉은빛이었다.
 하늘에 드리운 먹구름이 어찌나 밝은지 햇살처럼 환했다.
 소녀가 눈을 떴다.
```

이미지 Style transfer 모델

- GAN 기반의 Style transfer 모델인 `CycleGAN`을 사용
  - 프로젝트 초반에는 VGG를 이용한 **style transfer**를 사용하였다. 그러나, 사전 학습 모델은 입력받은 이미지에 대한 학습이 필요했다. 따라서 추론 과정에서 많은 시간이 소모됨.
  - 따라서 추론 과정에서 generator를 활용한 GAN 계열 모델을 탐색하기로 했다.
- CycleGAN
  - **Unpaired** Image to Image translation
  - 일반적인 GAN 모델은 paired dataset이 필요하다. 따라서 그에 따른 추가 이미지 수집이 필요하다.
  - 그러나 unpaired image dataset은 동화풍 이미지 수집만으로 학습이 가능하다. 즉 따로 label된 데이터셋을 만드는 과정 없이 사용 가능함. 데이터셋 구축에 시간이 많이 소요될 것으로 판단하여 CycleGAN을 선택함. 그리고 논문에서 GAN, GAN + forward, GAN + backward 와 비교했을 때 가장 성능이 좋은 것을 확인.



> Jun-Yan Zhu et al. , “Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks” , https://arxiv.org/pdf/1703.10593

1) 일반 이미지 → 동화 이미지 변환

2) 동화 이미지 → 일반 이미지

이슈 및 개선

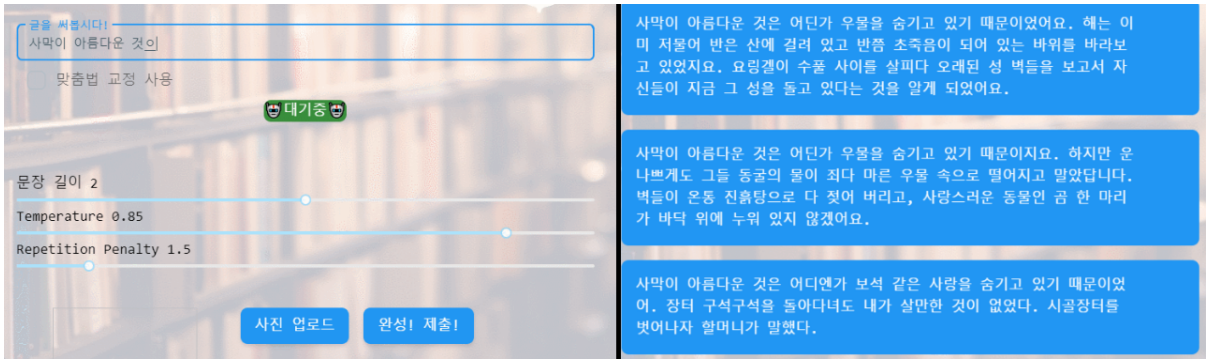
내부 테스트 및 프로토타입을 통해 프로젝트를 진행하면서 미처 고려하지 못 한 이슈들을 많이 발견하게 되었다. 기본적으로 `Github Issue`를 이용하여 이슈관리 및 개선작업을 진행하였다.

https://github.com/boostcampaitech3/final-project-level3-nlp-06/issues

- 문장 생성 모델의 이슈
  - 문장 어미가 통일되지 않는 문제
    - 데이터셋의 문체가 통일되지 않아서 일어나는 현상으로 파악. 데이터 전처리를 통해 문장의 문체를 통일하거나 더 많은 데이터셋을 확보하기로 결정
    - 실제로 더 많은 **데이터셋을 추가로 확보**하니 비교적 문체가 일정하게 유지됨
  - 등장인물의 비일관성
    - 짧고 다양한 동화 데이터셋을 이용한 만큼, 등장인물들이 너무나 많고 자주 바뀌게 됨.
    - 데이터 전처리를 이용해 데이터에서 등장인물들을 전부 `[MASK]` 토큰처럼 스페셜토큰으로 **마스킹**을 한 후, inference 이후 후처리 과정에서 사용자로부터 등장인물을 입력받아 교체하는 방식을 고민함
  - 따옴표, 쉼표 등 문장기호가 지나치게 반복되는 현상

        - 프로젝트 초반 데이터의 양과 품질이 낮아서 일어난 현상으로 파악함. 추가적으로 데이터를 확보하여 재학습 시킨 후 해당 현상이 해결됨
- 이미지 생성 모델의 이슈
    - 문장과 이미지의 연관성이 낮은 문제
        - 현재 프로젝트에서는 사용자로부터 이미지를 업로드받아 사용한다. 이런 경우 사용자가 쓰게되는 동화와 이미지간의 연관성은 매우 낮다. 완성된 글의 흐름을 미리 알기 어렵기 때문이다.
        - 따라서 DALL-E와 같은 **Text-to-Image** 모델이나, 이미지 캡셔닝과 같은 **Image-to-Text** 모델을 사용하는 방법으로 개선 제안

최종 결과물



자체 평가 의견

👍 잘한 점

1. 목표 설정 구체화 및 문서화

    지금까지의 프로젝트와 달리 최종 프로젝트는 과제가 정해진 것이 아니라 직접 문제를 정의하고 해결 방안을 구성하는 과정이었다. 때문에 구체적으로 무엇을 해야 하는지 `목표`를 설정하는 것이 아주 중요했다. 아이디어 도출부터 `문서화` 한 점이 좋았다. 이를 통해서 정확히 우리가 무엇을 하고자 하는지 명확하게 설정할 수 있었다. 주요 기능은 무엇이 있고, 어떤 기술을 사용하여 구현할 것인지, 일정은 어떻게 설정하고 협업 방식은 어떻게 할 것인지 등을 문서화해두니 의사소통 과정이 훨씬 효율적으로 이루어졌다. 프로젝트를 하면서 문서화 역시 아주 중요하다는 점을 익히 들어서 알고 있었으나, 이번 프로젝트 경험을 통해서 몸소 경험할 수 있었다.

2. 작업 분배

    각자의 능력과 일정에 따라 유동적으로 작업을 분배하고, 그때 그때 쏟아지는 이슈 및 추가 개발 사항을 쉼 없이 나누었다. 이 과정에서 일정이 딜레이되지 않도록 탄력적으로 배분을 했다. 예상되는 작업 시간의 50%를 여유로 붙이거나, 중간 결과를 보고하도록 하여 사고를 최대한 방지하려고 했다. 그럼에도 불구하고 결국 일정이 다소 지연되기는 했지만, 생각했던 최악의 상황보다는 훨씬 좋았다. 특히 누구 하나 소홀하게 되거나 너무 많은 작업을 하지 않도록 작업을 잘 분배한 것 같아서 그래도 계획한대로는 프로젝트를 진행할 수 있었다고 생각한다.

3. 어쨌든 하나의 큰 프로젝트를 무사히 끝냈다는 점이다. 우리 조는 다들 AI 관련 경험이 적어서 과연 잘 해낼 수 있을지 걱정이 많았다. 비록 계획한 모든 것을 다 해보지는 못 했지만, 그래도 한 80% 정도는 구현해서 뿌듯하

게 프로젝트를 마칠 수 있었던 것 같다. 확실히 5개월 전과 비교해보면 이제는 AI 프로젝트라고 하더라도 자신 있게 도전할 수 있을 정도로 성장했다고 생각한다.

💡 시도했으나 잘 되지 않은 것

1. 텍스트 생성 모델의 평가 지표
    - 생성 모델의 정량적인 평가가 부족했다. Perplexity와 같은 지표를 쓰긴 했으나, 그 외에도 멘토님의 조언에 따라 몇가지 아이디어를 더 시도하려고 했으나 잘 되지 않았다. 예를 들어, 한 가지 golden case를 두고 모델의 생성 결과와 비교하는 평가 지표가 있었는데, 결국 구체화하지는 못 했다.
2. 모델 경량화
    - 모델이 큰 만큼 Quantization을 적용하려고 했으나 생각보다 적용이 쉽지 않았다. 여러 버그들이 존재했다. data type을 통일해야 하는 문제, cpu / gpu 환경 이슈 등, 이를 해결하기에는 시간이 부족하여 결국 포기했다.

✏️ 아쉬운 점

1. 일정 관리
    - 일정이 타이트했기 때문에 미리 일정 계획을 유동적으로 잡았으나 실제로 작업에 들어가면서 생각보다 많은 이슈에 부딪히며 일정이 지연되는 일들이 많았다. 어느 정도는 지연되도 괜찮았지만 결국 작은 부분들이 누적되다보니 마지막 주가 될 즘에는 버그와 이슈만 고치다가 끝이 났다.
    - 프로젝트가 빠르게 진행이 되고 나면 기술 개선 및 교체하려 했으나 결국 시도하지 못 하고 계획으로만 남게 되었다. 프로젝트 서빙도 처음 하는 부분이다보니 강의 수강과 병행하느라 많은 시간을 쓰게 되었다.
    - 하지만 이제는 한번 씩 경험해보았기에, 다음 프로젝트를 시작한다면 이번보다는 훨씬 시간을 절약할 수 있을 것이라고 생각한다. 또, 어떤 문제들이 발생할 수 있을지 가늠을 어느정도 잡을 수 있게 된 것 같아서 효율적인 일정 관리를 할 수 있을 것이라고 느꼈다.
2. DALL-E 같은 멀티모달 모델을 시도해보지 못 함
    - 한국어 생성 모델인만큼 관련 데이터를 구하기 어려워서 포기했는데, 추후 여러 사례들을 조사해보니 모국어 → 영어 번역을 거쳐 DALL-E를 사용하는 케이스들이 여럿 있었다. (ruDALL-E 등)
    - 또, Inference 시간이 상당히 길어서 우리 프로젝트와는 적합하지 않는다고 판단하여 배제하였다. 하지만 시도는 해보는 것이 좋겠다고 생각이 뒤늦게 들었다. 그러나 프로젝트 일정의 문제로 결국 시도하지 못 하고 프로젝트가 끝나서 아쉬웠다.

개인 회고

강진희

학습 목표

- 스스로 문제를 정의하고 데이터를 수집, 분석과 모델링의 한 사이클을 도는 것
- 깃헙을 단순 레포 용이 아니라 협업 툴로 적극적으로 사용해보는 것
- 하나의 프로덕트를 온전히 완성해보는 것

프로젝트를 위해 시도한 방법

1) **데이터 전처리**

데이터를 수집했는데 수집한 데이터마다 형식이 달라서 전처리를 통해 형식을 맞춰주는 작업이 필요했다.

특히 이솝우화, 그림형제 동화 번역본은 작성자가 친절하게 단어에 대한 부연 설명을 괄호 안에 달아줬는데, 괄호 안 정보를 모두 제거하는 전처리가 필요했다. 그 밖에 설명 자료로 첨부한 링크도 정규 표현식으로 일괄 제거 했다.

가장 어려웠던 건 작성자의 첨언 부분. 글의 말미에 자신의 소감을 남겼는데, 이건 괄호 안 문자나 링크처럼 특별한 형태를 가지지 않아서 정규 표현식으로 본문과 구분하기 어려웠다. 그래서 직접 읽어가며 제거하는 수작업이 필요했다.

2) **키워드 추출**

프로젝트 말미에 서비스 플로우가 수정되었지만, 원래 GatherBook의 서비스는 이미지와 글에 연결성이 있었다. 완성된 글로부터 키워드를 추출하고, 키워드에 알맞는 이미지를 가져와 style-transfer하는 방식이었다.

하지만 키워드 추출을 포함한 이미지 방식의 번거로움, 적절한 이미지 검색 등의 이슈로 인해 서비스 흐름을 바꿨다.

키워드 추출은 BERT기반의 Key-BERT, TextRank, TF-IDF를 실험해봤는데 retrival이 아닌 extractive 방식에서는 TextRank 알고리즘이 가장 간결하고 빨랐다.

토크나이저는 Konlpy의 Okt(구 Twitter), Komoran을 사용해봤는데 inference에서 토크나이저 로딩과 명사 태깅이 시간을 가장 많이 잡아먹는 구간이었다.

이를 해결해보려 soynlp, kakaobrain의 pororo도 사용해봤지만 크게 다를 바는 없었다.

토크나이저 성능에 의존한다는 것, 토크나이징에서 시간이 오래 걸린다는 점을 감안해도 TextRank가 가장 빠른 방법이었다.

Key-BERT도 성능은 좋았지만 embedding에서 시간 소요가 매우 많았고, TF-IDF는 문서(동화) 케이스가 많아야 한다는 점과 키워드 추출을 할 때마다 축적된 문서 묶음 전체에 대해 inference를 찍어야 한다는 점에서 시간 소요가 많이 걸렸다.

3) **perplexity**

모델을 선택하는 로직을 위해 적절한 지표가 필요했는데 perplexity가 생성 모델의 성능 지표로 쓰인다는 피드백을 참고하여 perplexity를 사용해봤다.

깨달은 것

1) **데이터 양 vs 질**

비슷한 프로젝트의 레퍼런스를 참고해보니, 모델 단의 시도 보다는 데이터의 품질을 높여서 우수한 성과를 봤다는 이야기가 있었다.

그래서 수집한 데이터 중에서 길이가 길면서 기승전결이 뚜렷하다고 느껴진 신춘문예 당선작과 청와대 전래동화만 가지고 모델 학습을 시도했다. 하지만 결과는 크게 차이가 없었다.
수집한 데이터 중 절반만 가지고 사용했기 때문이라, 학습에 충분한 양을 확보하지 않은 문제라고 생각했다.

그리고 내가 생각했던 것보다 신춘문예 당선작과 청와대 전래동화의 품질이 그림형제 동화, 이솝우화와 크게 차이나지 않았던 것도 원인이라 생각한다. 100자 분량의 그림형제 동화 220편보다 500자 분량의 당선작 35편이 더 나을 것이라 생각할만큼 품질의 차이는 없었던 것 같다.

2) **점수를 위한 모델링과 서비스를 위한 모델링**

서비스를 기획하면서 최종 모델을 선택할 때 가장 큰 영향을 미친 건 inference 속도이다. 사용자가 직접 서비스를 이용할 때, 만족감을 해치지 않을 time limit을 지켜야 한다는 것이 중요했다.

키워드 추출 단에서도 알고리즘 선택에서 가장 중요하게 생각한 것이 inference 시간. 특히 생성모델과 다르게 키워드 추출은 서비스의 가장 중심적인 기능이 아니었고, 그렇기 때문에 inference 시간이 길어서는 안됐다.

Key-BERT가 아닌 TextRank를 선택한 이유도 출력 시간 때문인데, 비슷한 결과를 보이면서 동시에 가장 빠른 inference 시간을 보인 TextRank를 선택했다.

추가로 고려했던 TF-IDF는 동화라는 소재가 비슷한 키워드 노출이 잦을 수 있어서 적절한 방법은 아니다 라는 생각이 들었다.

3) **생성 모델의 창의성은 어떻게 표현할까?**

perplexity는 생성 모델이 작성한 문장이 얼마나 높은 확률을 가지고 있는지를 나타내는 지표라고 이해했다. 하지만 기획한 서비스는 창작 서비스이고, 창작에서 가장 중요한 건 독창적인 아이디어라고 생각하는데, 이걸 적절하게 표현할 지표가 있을까 생각했다.

새롭게 시도한 변화와 효과

Github issue를 적극 활용했고, pull request를 사용해봤다.
처음에는 팀 레포에 코드를 덮어씌우지는 않을까, 자료가 날아가지 않을까 엄청 무서웠다. 그래서 브랜치 파놓고 계속 실험하다가 commit 하기 전에 내용을 지우기도 했다. 100을 썼다면 30만 커밋하는 수준. 그래도 브랜치 파서 실험과 기능 개발 해보고 pull request해봐서 만족한다.

깃헙 이슈로 이슈에 대한 진행사항까지 같이 파악할 수 있어서 좋았다. 이전에는 노션을 주된 프로젝트 협업 툴 내지는 소통 창구로 썼는데, 코드 작업하면서 사용하는 깃헙에 이슈 파악까지 기능을 추가하니 훨씬 편했고, 따로 노션에 들어갈 필요가 없어서 번거로움이 줄었다.

마주한 한계와 아쉬웠던 점

백엔드를 솔직히 시도해보고 싶었는데 역량이 안되어서 포기했다.

당장 프로젝트 시간은 촉박하고, 내 개인 공부를 위해 프로젝트의 역할을 맡았다가 망치면 어쩌나 하는 생각이 들었는데, 고장난 손이라도 보탰다면 도움이 되지 않았을까 하는 아쉬움이 있다.

역할을 맡고 성장하기 위해서는 도전하는 용기가 필요하다고 생각한다. 그 점이 이번 프로젝트에서 가장 아쉬운 부분이다.

차경민

학습 목표

- 데이터 제작부터 AI 프로덕트 서빙까지 직접 경험하는 것
- Github을 이용하여 다른 팀원들과 의견을 공유하며 협업을 진행하는 것
- 프로젝트 온전히 집중하여 몰두하는 것

프로젝트를 위해 시도한 방법

1. 데이터 수집과 전처리

   프로젝트를 수행하기 위해서는 동화 텍스트 데이터와 이미지가 두가지가 필요했다. 동화 이미지는 저작권 문제가 없는 이미지를 수집하려고 노력하였고, 기존 논문에 있는 데이터셋도 이용하여 학습을 진행하였다. 텍스트 데이터의 경우 그림형제, 국립국어원, 청와대 전래동화, 신춘문예 등을 수집할 수 있었고 텍스트 모델을 계속해서 발전시키는 과정에서 텍스트 데이터의 품질의 중요성을 알 수 있었다.

2. CycleGAN

   사용자가 업로드한 이미지를 동화풍으로 변환해주기 위해 style-transfer에 대한 자료들을 찾아보았다.

2-1) pretrained 네트워크를 이용하는 방법

pretrained된 모델을 기반으로 content image와 style image을 입력으로 이용해 이미지를 학습하는 방법으로 결과를 뽑아보았다. 이미지 2장으로 style transfer가 된다는 점과 style image에 따라 퀄리티 좋은 동화풍 이미지가 생성된 것을 확인했지만 매번 새로운 이미지가 들어올 때마다 매번 이미지를 새롭게 최적화 해야 하므로 inference 시간이 오래 걸린다는 문제가 있었다. (30~40초)

2-2) style transfer network를 학습시키는 방법

위의 inference 시간이 길다는 문제점을 해결하기 위해 직접 모델을 학습시키는 방법에 대해서 자료를 찾아보았다. 자료를 찾아보니 GAN에 대한 자료들을 찾아볼 수 있었고 그 중에서도 CycleGAN을 실험해보기로 했다. 그 이유는 프로젝트의 시간 제약으로 인하여 모든 이미지에 대해서 paired 된 동화 이미지를 labeling하는 작업을 하지 않는 unpaired image to image translation이라는 점과 inference 과정에서 한 번의 feed forward 네트워크만 거치기 때문에 inference 시간이 짧다는 점에서 Cyclegan을 사용하기로 하였다.

X 도메인 이미지를 일반적인 풍경 사진, 사람 사진으로 두고, Y 도메인 이미지를 동화풍 이미지로 두었고 X 도메인 이미지를 Y 도메인 이미지로 바꿔주는 네트워크 G:X->Y, 그리고 그 반대로 바꿔주는 네트워크 F:Y->X를 동시에 학습하였고 cyclegan의 목접함수도 논문에 제시된 것처럼 기존의 GAN의 Loss에서 domain transfer 후에도 같은 content를 유지하도록 cycle-consistency loss를 추가하여 학습을 진행하였다.

3. 텍스트 모델

   3-1) RNN 기반의 언어 모델

   Seq2seq를 구현하기 위해 기존의 언어 모델부터 구현을 할 생각으로 RNN 기반의 언어 모델을 구현하였다. 동화 데이터 셋으로 vocab을 구축하고 lstm, gru를 이용한 언어 모델을 만들어 보았지만 역시 전혀 알아볼 수 없는 문장이 계속 생성된 것을 확인할 수 있었다. 시간 제약상 해보지 못한 것이 많아서 살짝 아쉬웠던 작업 중에 하나이다. vocab을 띄어쓰기 단위가 아닌 Subword tokenizer를 사용하는 것, seq2seq 구현 등을 시도하고 싶었지만 시간 제약상 제한이 있었다.

   3-2) KoGPT

   KoGPT-trinity, KoGPT2를 이용하여 학습을 진행하였다. 생성된 결과가 문장 단위로 나오는 것이 아닌 중간에 짤려서 나오는 것에 대한 문제를 해결하기 위해 문장 앞뒤로 스페셜 토큰을 더해줌으로써 문장 단위 생성을 진행할 수 있었다. 또한 기존에 동화의 구분 없이 데이터를 feed 해주던 방법을 동화 단위로 학습을 진행할 수 있도록 코드를 수정하였다. 그 결과, 몇가지 실험에서 이전보다 훨씬 더 문맥상 어울리는 문장을 생성하는 것을 확인할 수 있었다.

4. Perplexity

   언어 생성모델의 성능 지표로 가장 보편화된 perplexity를 구현하였다. 기존 Trainer 클래스를 상속하여 compute_metrics와 compute_loss에 대한 오버라이딩 함수를 구현하려고 시도했지만 huggingface 라이브러리에 대한 이해 부족과 시간에 쫓기어 오버라이딩 함수를 구현하는 것까지는 실패하였다. 하지만 모델이 생성한 결과에 대한 perplexity 함수는 구현하여 inference 할 때마다 perplexity 값을 계산할 수 있도록 구현하였다.

깨달은 것

트랙이 NLP여서 텍스트 데이터와 모델에 대해서 많이 접해보았지만 이미지 관련 모델 그것도 GAN을 경험한 것은 이번이 처음이었다. 처음이여서 조금 낯설긴 했지만 논문 내용과 다양한 블로그 글을 참고하여 코드를 구현할 수 있었다. GAN을 경험하면서 NLP와 CV 모두 input 데이터만 달라지는 것뿐, 데이터 셋과 데이터 로더, 모델, loss 함수를 정의하고 개선하는 과정은 크게 다르지 않다는 것을 느낄 수 있었다.

새롭게 시도한 변화와 효과

문제가 있을 때마다 Github Issue를 활용하였고, 모델 개발, 새로운 feature를 개발할 때마다 새로운 브랜치를 생성하여 작업을 진행하였다. 처음에는 조금 번거로운 점이 있었지만 코드를 공유하고 같이 issue를 공유하는 점이 좋았고 한 눈에 어느 코드를 수정했는지 볼 수 있어서 편리했다.

마주한 한계와 아쉬웠던 점

Dalle를 시도해보지 못한 것. 시간적으로 여유가 없기도 했지만 최근에 김성훈 대표님께서 dalle동화에 대한 프로토 타입을 제작하신 것을 보고 시도해보지 못한 부분에서 많이 아쉬움이 남았다.
백엔드를 진행하려고 했는데 그러지 못한 것. Product serving 강의에 나온 FastAPI를 이용하여 시간을 조금 투자해보았다면 백엔드를 만들었을 텐데 중간에 번아웃도 오고 선재님이 더 잘할 것이라 판단이 되어 진행하지는 않았다. 백엔드 전반을 경험해보지 못한 점이 살짝 아쉽다.

김선재

학습 목표

- AI 기술을 접목한 아이디어를 생각하고 서비스를 설계해보자
- 단순하더라도 한 사이클을 완성시켜보자
- 강의에서 배운걸 기반으로 실제 프로덕트 서빙 경험을 쌓자

프로젝트를 위해 시도한 방법

1. Markov-Chain

처음부터 AI 모델을 사용하는 것보다 단순하게 한 사이클을 만들어보고자 단순한 문장 추천, RNN, LSTM 모델을 찾다 발견해 모델 구현을 시도했다. 단순하면서 강력한 모델이였지만 OOV 문제에서 굉장히 취약한 모습을 보였다. 또한 현재 데이터베이스에 있는 문장을 기반으로 한 확률 모델이였으므로 하나의 입력에 대해 여러 결과물을 내보낼 수 없다는 점이 아쉬웠다. 해당 문제를 해결하기 위해 MCMC 방식을 도입해보려 시도했으나 레퍼런스와 시간 부족으로 인해 모델을 완성 시키지 못했다.

2. 데이터 전처리

처음 GPT 모델을 사용했을때 별도로 BOS, EOS 토큰을 사용하지 않은 채로 모델을 학습시켰다. 학습시킨 모델에서 생성된 문장들은 문맥이 부자연스러워 BOS, EOS 토큰을 별도로 추가했다.

문장을 생성할때 어떤 경우 아무런 output없이 줄바꿈만 여러개 출력되는 경우가 종종 발생했다. 이를 해결하기 위해 나뉘어져 있던 텍스트 파일을 하나로 모으고 모든 빈 줄들을 삭제한 후 한 문장 끝 이후 줄바꿈을 추가했다.

한국에 문장 생성을 더 용이하게 하기 위해 모든 알파벳, ㅋㅋ,ㅎㅎ,ㅠㅠ 등의 한국어 slang들을 제거해 주었다.

3. 백엔드

강의에서 들은 FastAPI를 활용해 백엔드를 구축했다. 복잡하게 연결된 테이블을 사용하지 않을것이기에 NoSQL인 MongoDB를 사용해 데이터베이스를 구축했다. 처음에는 핵심 기능만 작동하게 구현한 후 에러 발생 가능 선택지를 생각하고 이를 try,except 또는 if, else문으로 예외처리를 하며 함수를 생성했다.

깨달은 것

1. 데이터의 양질

많은 데이터를 모으고자 그림형제 번역본을 모델에 학습시켰을 때 모델이 생성하는 문장이 부자연움을 깨달았다. 이는 번역을 하며 발생한 독일식 명사 표현과 번역 문장 종결 어미의 어색함으로 원인을 파악했다. 다양한 데이터별

로 학습시키고 생성 결과를 비교해 보며 다시 한 번 데이터의 양질의 중요성을 깨달을 수 있었다.

2. 백엔드 구축 단계

백엔드, 프론트엔드 구축 전, 페이지 명세서를 같이 만든 후 함수를 구현했다면 백엔드를 구현하는데 덜 헤맸을거란 생각이 들었다. 이번 프로젝트에서는 시간 관계상 프론트엔드, 백엔드를 동시에 진행했다. 백엔드를 처음 구축하며 나는 모든 기능을 세부적으로 나눠 각각 함수로 구현했다. 이후 프론트와 연결하기전 굉장히 비효율적인 코드였음을 깨달을 수 있었고 코드 리팩토링을 하는데에 많은 시간을 소요했다.

마주한 한계와 아쉬웠던 점

부스트캠프 전체 과정을 걸어오며 계속 느꼈던 한계점은 스스로의 지식 습득 역량이였다. 제공된 교육, 인적 인프라 퀄리티에 비교해 습득하고 내 것으로 온전히 가져간 지식이 기대보다 너무 적어 5개월의 시간동안 조금씩 지쳐가면서 내 자신에게 항상 화가 나 있었다. 비슷한 감정을 느낀 캠퍼분들이 감정을 슬랙에, 그리고 스페셜 피어세션 때 가감없이 공유해 주셨기에 덜 외롭게 감정을 느낄 수 있었다.

마지막 프로젝트를 참여하며 아쉬웠던 점 하나는 깃에 코드 푸쉬를 많이 못 한 것이다. 해당 프로젝트에서 데이터 수집/처리와 서버, 그리고 백엔드를 담당해 코드를 올릴 일이 많이 없었다. 타 캠퍼 분께서 처음 소개해주신 이슈 공유, pull request, git convention을 한번 여러번 시도하고 배워보고 싶었는데 아쉬웠다.

김태훈

학습 목표

- 아이디어를 기반으로 AI 프로젝트를 기획하고, 프로덕트 서빙까지 전체 사이클 경험
- 협업 툴을 적극 사용해 실제 현업과의 유사한 프로세스를 따라가 보는 것
- 이론에서 벗어나 실무에서 필요한 고민과 스킬들을 체화해보는 것

프로젝트를 위해 시도한 방법

1. 데이터 수집 및 전처리

   기본적으로 프로젝트 수행을 위해서는 텍스트 데이터와 이미지 데이터 두 가지가 모두 필요했기 때문에, 다양한 데이터를 수집해야 했다. 동화 번역본의 경우 역자가 추가적으로 넣어둔 정보들을 전처리 해주었고, 링크 등 내용 외의 텍스트가 다수 포함되어 있어 제거해주는 작업을 거쳤다. 자동으로 필터링하기 어려운 텍스트들도 있어, 일부는 직접 수작업으로 처리하기도 했다. 그밖에 한국어로 된 SNS 표현들을 제외해 주었고, 영어 표현과 같이 내용과 크게 상관없는 부분도 처리해 주었다.

3. 모델링

   RNN 기반의 언어 모델, GPT 계열, BERT 계열 등 이번 최종 프로젝트에서는 가용 가능한 모든 언어 모델이 후보군에 있었다. 실제 동화 데이터로 Vocabulary 를 만들고, 각 모델들의 성능을 비교해보는 과정에서 RNN 기반의 언어 모델은 생각보다 결과가 좋지 않아, 제대로 구현한 것이 맞는지 확인하는 과정을 여러 번 거쳤다.

   프로젝트의 시간 제한상, 더 다양하고 많은 시도를 해보지 못한 것이 매우 아쉽지만, 언어 모델 중에서 여러 방향의 프로세스를 겪어본 것은 좋은 경험이라고 생각한다. 예를 들어 키워드를 추출하는 부분, 그리고 이 과정에서 전통적인 모델들부터 SOTA까지 성능을 비교해 보았던 부분, 글의 일부가 주어지면 다음에 나올 문장을 생성해보는 부분 등 부스트캠프 과정 전체에서 배웠던 내용들을 한데 묶어서 실험해보는 듯한 과정을 거쳤다.

   생성 모델을 사용할 때는, 생성된 결과가 온전한 형태로 나오지 않는 문제에 봉착해 스페셜 토큰을 사용해 해결했다. perplexity 기준으로 수치가 더 좋으면 무조건 더 좋은 결과가 나오는 것은 아니라는 사실을 실제로 체험

해 보았고, 동화 학습에 있어 보다 용이하게 진행할 수 있는 전체적인 파이프라인을 구축해볼 수 있는 시간이었다.

깨달은 것

생성 모델을 다뤄보면서, 단순히 언어 모델만 건드리는 것이 아니라 이미지와 연결되는 프로세스를 겪어본 것이 정말 값진 경험이라고 생각한다. 실제로 진행해보기 전에는 겁을 많이 먹었는데, 코드 단에서 살펴보니 도메인만 다를 뿐, 꽤나 공통적인 부분이 많아 기반이 되는 공부의 중요성을 다시금 느꼈다. 또한 스페셜 토큰의 중요성처럼, 이론적으로 배운 것을 실제로 직접 겪어보는 과정의 가치 역시 직접 체험할 수 있었다.

새롭게 시도한 것

이전보다 Github issue를 훨씬 적극적으로 사용했고, Pull request 를 써보았으며, feature 단위로 새로운 브랜치를 도입해서 작업을 진행했다. 처음에는 익숙하지 않았지만, 수정 기록을 확인할 수 있다는 점이 매우 편리하고 따로 정리할 필요성이 줄어들어 앞으로는 이전보다 더 익숙하게 사용할 수 있을 것 같다.

마주한 한계와 아쉬웠던 점

맡은 일 이외의 부분들을 더 많이 경험해보지 못한 것이 아쉽다. 물론, 협업에서는 각자의 역할이 있는 것이겠지만 조금 더 배려하면서 서로 도울 수 있는 부분도 있는데, 보다 적극적으로 팀원들을 돕지 못한 것이 매우 아쉽다.

개인적으로는 한 발 더 나아가지 못한 모든 부분들이 아쉽다. 모델링 파트에 있어서 Dalle 기반의 모델까지 시도해보지 못한 것, 오프라인/온라인 설문을 진행할 때 더 많은 사람들, 더 다양한 연령대의 소리를 듣지 못한 것, 백엔드 파트에 더 깊이 관여하지 못한 것 등 아쉬운 부분이 정말 많다. 그러나 한편으로는, 이와 같이 아쉬운 퍼포먼스에도 함께해주며 마무리까지 완벽하게 해준 팀원들에게 정말 감사한 마음이 드는, 감사한 경험이었다.

능력의 부족을 정말 많이 느꼈고, 그렇기에 더 정진할 수 있는 동기를 얻은 것 같다. 남에게 진정한 도움이 되기 위해서는 실력을 열심히 쌓아야겠다는 생각을 여러 번 하게 된 경험이었다.

이도훈

학습 목표

- 자유 주제로 AI 프로젝트 경험을 해보는 것
- 모델링 및 학습부터 프로덕트 서빙까지 개발의 한 싸이클을 경험

프로젝트를 위해 시도한 방법

1. KoGPT 파인튜닝

   생성 모델의 대표인 GTP를 써보는 것은 처음이었다. GPT의 학습 방식은 다음 토큰을 예측하는 것인데, 우리가 하고자 하는 task 역시 다음 토큰을 예측하는 것으로 GPT의 pre-train task와 fine tuning task가 같다. 따라서 그리 어렵지 않게 파인튜닝을 마칠 수 있었다.

2. 모델 성능 향상

   먼저 파인튜닝을 마친 상태로 바로 쓰는 경우에는 여러 문제들이 발생했다. 특정 토큰 이후에 `발산`하여 계속 같은 토큰이 반복되거나 의미 없이 이어지는 현상을 겪었다. 기본 설정으로 Greedy Search를 하기 때문이었다. 이후 확률에 기반하면서도 자연스럽게 생성을 이어나가는 여러 방법들을 배우게 되었다. 가령 Beam search 처럼 여러 시퀀스에 걸쳐 높은 확률을 갖는 시퀀스를 찾는 방법과 Samlping이었다. 확실히 Greedy search 보다는 훨씬 자연스러운 생성을 보여줬으나 여전히 동어 반복등의 문제는 발생했다. 하지만 최선의 아웃풋을 찾는 다양한 노력들을 보면서 앞으로 어떤 새로운 방법이 등장하더라도 잘 따라갈 수 있을 것 같았다.

생성과 별개로 모델의 성능 평가 지표도 주요 관심사였다. 파인 튜닝 후 실제 출력의 성능을 정량적으로 평가하기에는 마땅한 지표가 없었다. perplexity와 같은 지표를 보조적으로 활용할 수는 있었지만, 여전히 모자란 느낌이었다. 멘토링을 거치며 몇가지 아이디어를 제공받았지만, 구현하는데는 실패해서 많은 아쉬움이 남았다.

깨달은 것

주어진 것 하나 없이 정말 자유롭게 시작한 주제다보니 우선 데이터를 구하는 것부터가 난관이었다. 다행히도 저작권에서 비교적 자유로운 데이터들을 찾을 수 있었고, 양도 어느정도 확보되었다. 약 4MB 정도가 되었는데, 이정도로 파인튜닝을 하여도 모델이 꽤 그럴듯하게 흉내를 내는 것을 보며 기쁨과 함께 더 많은 데이터가 들어간다면 어떻게 될지 궁금하기도 했다. 또 perplexity 처럼 이론으로만 보았던 것들을 프로젝트와 연결시켜 수치를 살펴본 경험도 좋았다. 이론을 실제로 검증한 느낌이었기 때문이다. 그밖에도 깨달은 점은 데이터의 중요성이었다. 같은 데이터라고 하더라도 전처리를 어떻게 하는지, batch 단위를 어떻게 하는지에 따라 모델의 성능이 바뀌는 점 또한 데이터에 대한 시각을 넓히는데 도움이 되었다.

새롭게 시도한 것

서빙까지 고려하여 코드를 작성해야 해서 확실하게 코딩 컨벤션과 Github의 기능들을 본격적으로 이용하자고 했다. 특히 각자 맡은 부분은 branch를 따로 파서 merge 충돌을 최대한 방지하고자 하였고 덕분인지 프로젝트를 진행하면서 충돌이 난 적은 상당히 드물었다. 또, 각자 맡은 기능 구현을 바로 merge하기 전에 반드시 코드 리뷰를 거치도록 하였는데, 이를 통해서 다른 팀원들도 해당 코드가 무슨 기능을 하고, 어떻게 사용하는지 숙지하게 되어 회의시간을 더 효율적으로 사용하게 된 것 같다. 그때 그때 PL 때문에 기록을 남겼기 때문에 추후에 문서화하는데도 도움이 되기도 했다.

마주한 한계와 아쉬웠던 점

처음 계획을 세웠을 때는 목표를 조기에 달성하면 StyleGAN을 DALL-E로 치환하고자 하였는데 결국 여러 이슈들 때문에 일정이 딜레이가 되고 손을 대지 못 해서 아쉬웠다. 그 밖에도 해보고 싶은 것은 많았으나 시간의 한계상 시도하지 못 한 것들이 너무 많았다. 경량화, 데이터의 추가적인 전처리, 인물 마스킹을 통해 등장인물을 고정시키는 방법 등등… 아무래도 5주간의 시간 내에 강의와 함께 병행해야 하다보니 시간이 정말 많이 부족했다. 체감상으로는 약 3주만에 한 느낌이었다. 또 프로젝트 후반에는 프론트엔드 작업에 집중하게 되느라 모델을 많이 건드리지 못했는데 이부분도 많이 아쉬웠다. 맡은 역할이 있으니 어쩔 수 없으나, 대충 끝내고 모델에 더 집중할 걸 그랬나 하는 아쉬움이 들었다.

그 밖에도 여전히 배울것이 많다는 점을 느꼈다. 5개월간의 부스트 과정이 끝났지만, 이제 끝이 아니라 시작이라는 느낌을 강하게 받았다. 하지만 이전과는 다르게 그래도 배우려고 한다면 배울 수 있는, 그런 기본 자격쯤은 갖추지 않았나 하는 자신감도 생겼다.