랩업 리포트

1. 프로젝트 개요

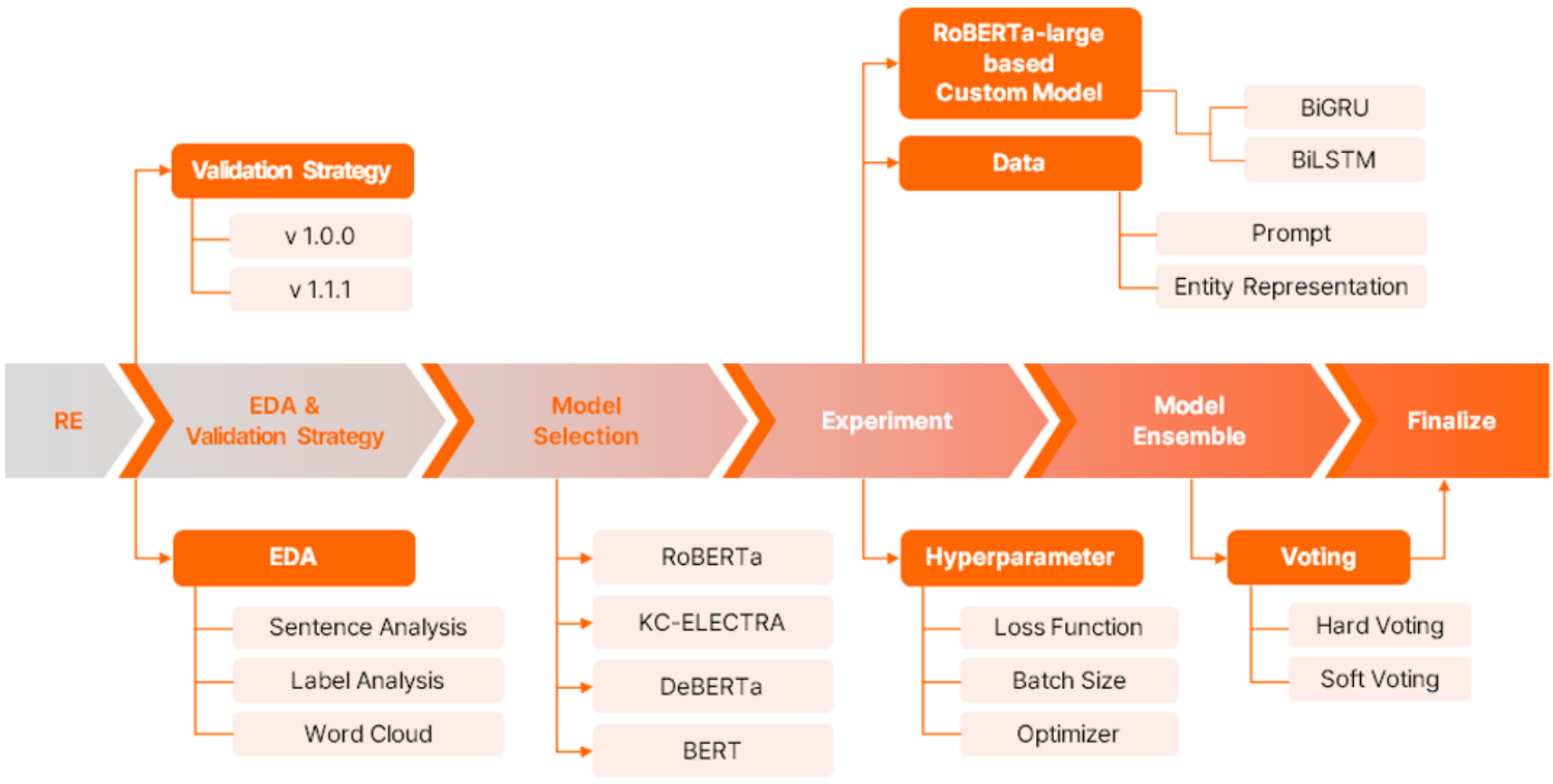
일정

23. 05. 03. (월) 10:00 ~ 2023. 05. 18. (목) 19:00

주제

문장 내 개체 간 관계 추출 — 문장의 단어(Entity)에 대한 속성과 관계를 예측하는 모델 만들기

프로젝트 flow



데이터

KLUE(Korean NLU Benchmark) RE(Relation Extraction) dataset

- train : 32,470 개
- test : 7,765 개
- 구성

id	sentence	subject_entity	object_entity	label	source
0	0	〈Something〉는 조지 해리슨이 쓰고 비틀즈가 1969년 앨범 《Abbey Road》에 담은 노래다.	{'word': '비틀즈', 'start_idx': 24, 'end_idx': 26, 'type': 'ORG'}	{'word': '조지 해리슨', 'start_idx': 13, 'end_idx': 18, 'type': 'PER'}	no_relation

- label

```
{'no_relation': 0, 'org:top_members/employees': 1, 'org:members': 2, 'org:product': 3, 'per:title': 4, 'org:alternate_names': 5,
'per:employee_of': 6, 'org:place_of_headquarters': 7, 'per:product': 8, 'org:number_of_employees/members': 9, 'per:children': 10,
'per:place_of_residence': 11, 'per:alternate_names': 12, 'per:other_family': 13, 'per:colleagues': 14, 'per:origin': 15,
'per:siblings': 16, 'per:spouse': 17, 'org:founded': 18, 'org:political/religious_affiliation': 19, 'org:member_of': 20,
'per:parents': 21, 'org:dissolved': 22, 'per:schools_attended': 23, 'per:date_of_death': 24, 'per:date_of_birth': 25,
'per:place_of_birth': 26, 'per:place_of_death': 27, 'org:founded_by': 28, 'per:religion': 29}
```

평가방법

- Micro F1 score
  - 정답 no_relation을 정확히 예측한 경우를 제외한 모든 사례에 대하여 micro F1 score 측정에 포함
  - Micro F1 score의 경우, 전체 데이터의 FP와 FN 그리고 TP를 구한 뒤 TP와 FP로부터 precision을, TP와 FN으로부터 recall을 구하여 precision과 recall의 조화평균을 취하여 구함

$$\text{Micro } F_1 = (\text{precision} \| \text{recall})$$

  - 다중 클래스 분류(multi-class classification) 문제에서는 정확도(accuracy)와 동일

- AUPRC
  - 30개의 클래스에 대한 다중 클래스 문제를 가정
  - Threshold를 0부터 1까지 값을 연속적으로 변화시켜가며 각 클래스에 대한 예측 확률이 담긴 30차원의 tuple의 각 요소와 threshold의 대소 관계를 비교하여 해당 클래스를 positive로 예측할지 말지 여부를 결정하면서 precision과 recall 값 변화를 관찰. 즉 precision과 recall은 각각 threshold를 매개변수로 하는 함수이고, precision vs. recall 그래프를 2차원 평면에 그렸을 때 그래프 아래 면적 넓이가 AUPRC 값.

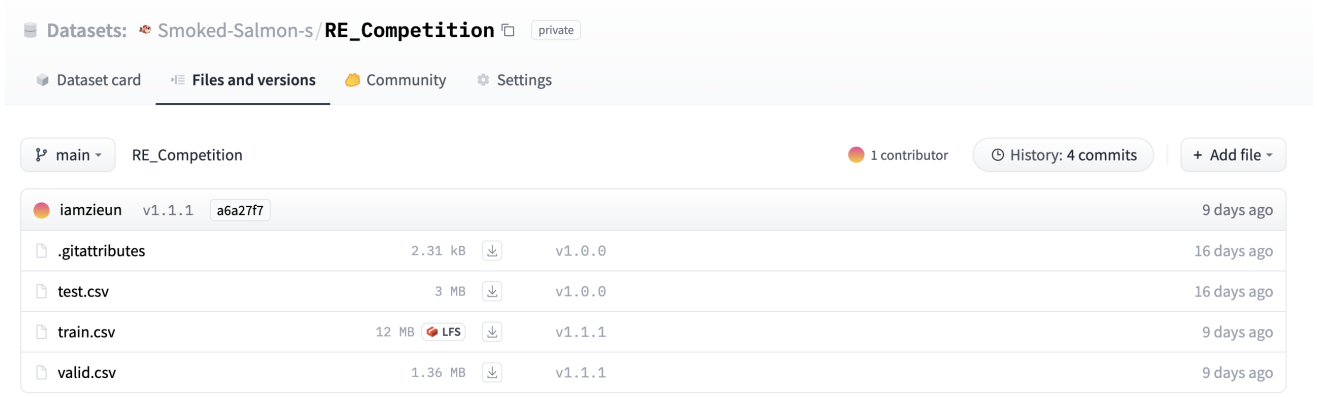
구성 및 역할

공통	EDA 공유, 모델 실험
문지혜	loss function 구현 및 실험, PLM 모델 실험
박경택	프로젝트 전반부 PM, custom model 구현 및 구조 변화 실험
박지은	main code refactoring, 데이터 버저닝, prompt 구현 및 실험, custom model 실험 및 사후 분석
송인서	프로젝트 후반부 PM, main code refactoring, Sweep 코드 작성
윤지환	entity marker, TAPT, embedding layer 구현, 학습결과 사후분석, 앙상블

협업

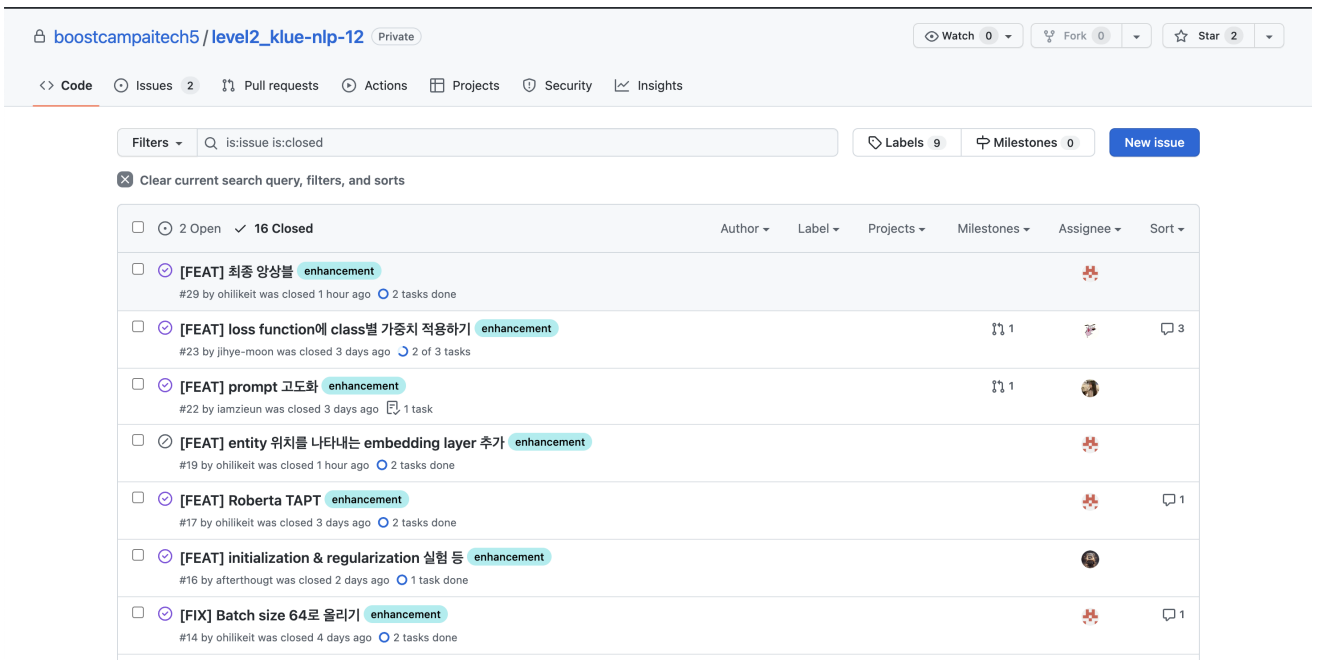
데이터 버저닝 - HuggingFace Datasets

- Semantic Versioning 방식의 데이터 버저닝



코드 버저닝 - GitHub

- GitHub flow 방식의 코드 협업
  - main 브랜치 하나를 운용하여 빠른 기능 개발 및 지속적인 통합이 가능한 협업 방식
  - 코드에 새로운 기능 추가 시 Issue 추가 후 main에 병합하기 전 pull requests를 통한 코드 리뷰



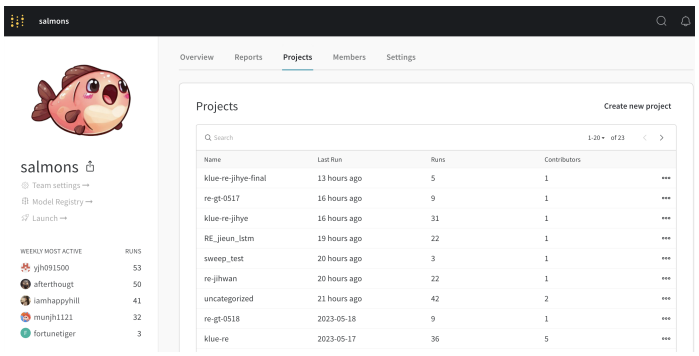
- 커밋 메시지 규약

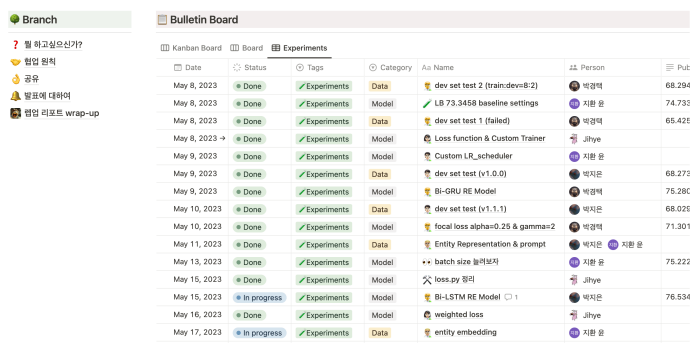
  참고: https://www.conventionalcommits.org/ko/v1.0.0/

  (Angular.js 커밋 가이드라인에 큰 영향을 받음)

실험 관리 및 기록 — WandB & Notion

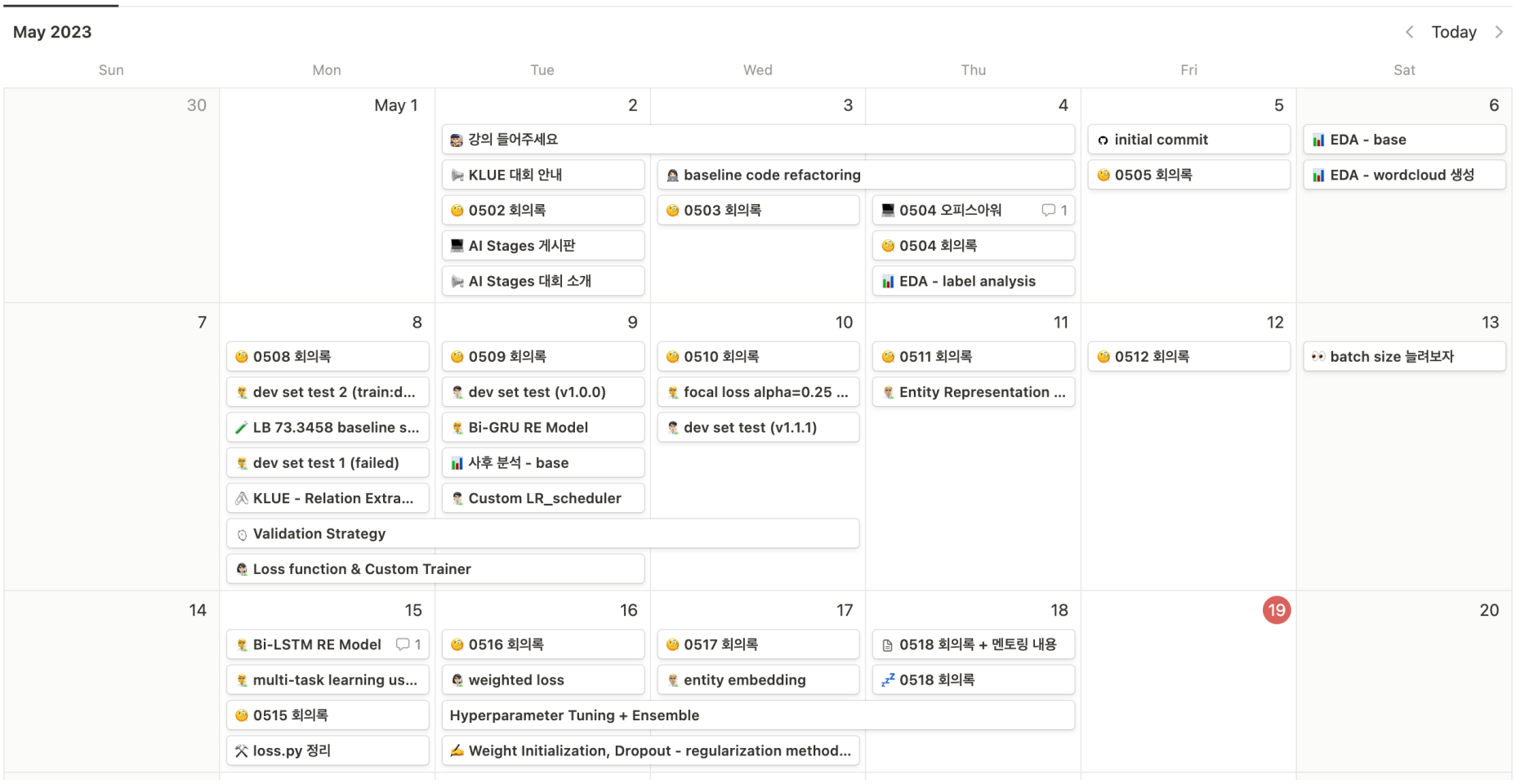
- WandB 플랫폼을 이용하여 실험 관리
- Notion을 통해 자세한 실험 내용 및 결과와 해석, 토의 등을 정리하여 기록 및 내용 공유





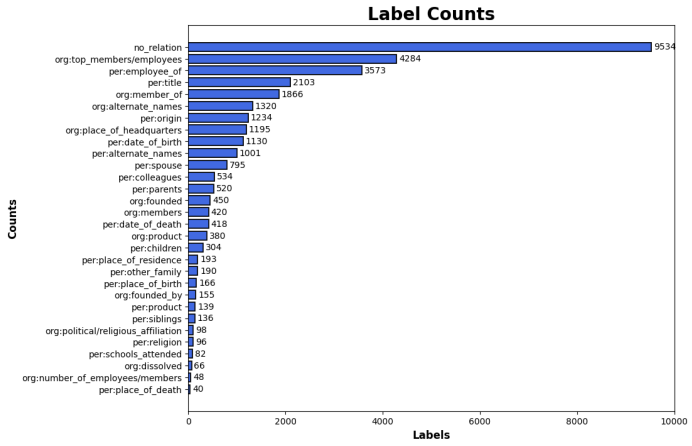
2. 프로젝트 과정

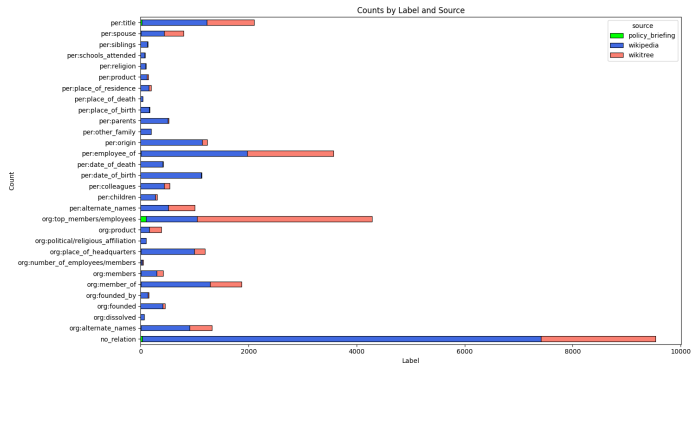
타임라인



EDA

- label 분포에 imbalance가 보인다.
  - `no_relation`(관계 없음)이 압도적으로 많고 `org:top_members/employees`와 `per:employee_of`가 뒤를 잇는다.
- 출처별로 각 라벨을 구분해보면 wikipedia가 가장 많다.

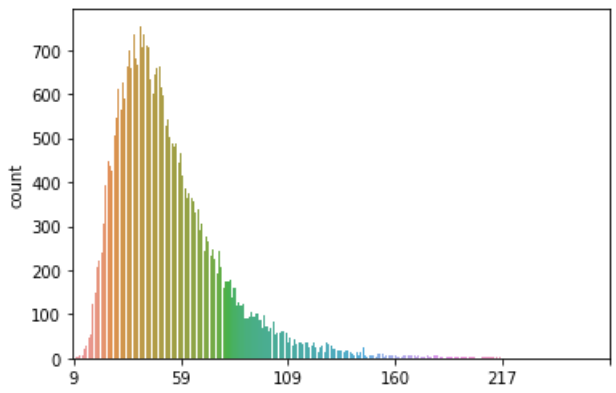


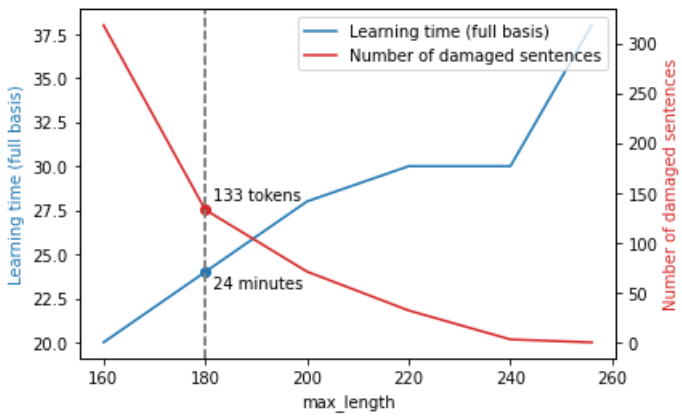


- input 중복여부를 확인해보면,
  - 같은 문장의 entity 종류와 관계를 바꾼 경우가 꽤 많았음.
  - 5개의 label이 이상한 example들을 더 적합한 label을 남기고 아예 똑같은 예제들은 하나만 남기고 제거(id = [6749, 7276, 8364, 22258, 4212])

	sentence만	sentence + obj_entity	sentence + sub_entity	sentence + obj + sub	다 같고 라벨만 다른 경우
겹친 문장 수 (개)	3,667	877	1,263	47	42

- 문장 당 token 갯수는 대부분 180개 이하이고 max_length 증가에 따른 학습시간 및 sentence 손상 갯수를 고려하여 tokenizer의 max_length를 180으로 설정





학습 기본 세팅

1) Validation set

1. stratifed 방법으로 train:valid = 8:2 비율로 분리
    a. valid f1 score가 85점이 나오나 리더보드 점수는 72점정도로 큰 차이를 보임
    b. valid data가 test data와 큰 차이를 보여서 다른 방법을 고려함
2. train set의 라벨 분포와 학습 후 inference 결과의 라벨 분포가 큰 차이를 보임
    a. train set : `no_relation`(29%) —> inference output : `no_relation`(60%)
3. test set의 분포를 알 순 없으므로 신뢰할 수 있는 수준(`LB score 73`)으로 학습시킨 모델 선정
    a. test 데이터에 대한 inference 결과 나온 분포와 유사하도록 각각 train : valid = 9 : 1로 분리
4. valid f1 score가 0.75 나올 때 LB score가 0.73정도로 둘 사이의 간극이 줄어들었고 신뢰할만한 valid set을 얻음

2) Base model

- KLUE 데이터로 pre-train 시킨 `klue/roberta-large` 모델을 실험을 위한 기본 모델로 설정
- `klue/bert-base`, `KcELECTRA`, `mdeberta` 와 같은 모델들도 실험해보았으나 KLUE 데이터로 이미 사전학습을 거친 `klue/roberta-large` 모델에 성능이 훨씬 못미침

Model	Explanation	Micro F1 Score	AUPRC
klue/roberta-large	KLUE Dataset으로 RoBERTa 모델을 사전학습한 모델	**71.371**	**69.653**
klue/bert-base	KLUE Dataset으로 BERT 모델을 사전학습한 모델	66.751	64.76
beomi/KcELECTRA-base-v2022	네이버 뉴스에서 댓글과 대댓글을 수집해, 토크나이저와 ELECTRA 모델을 처음부터 학습한 Pretrained ELECTRA 모델	61.722	46.494
lighthouse/mdeberta-v3-base-kor-further	microsoft 가 발표한 mDeBERTa-v3-base 모델을 약 40GB의 한국어 데이터에 대해서 추가적인 사전학습을 진행한 언어 모델	65.571	55.646

실험 - Input 변화

1) Prompt

- baseline code에서 문장 앞에 존재하던 `[CLS] sub_entity [SEP] obj_entity [SEP]` prompt가 성능향상에 큰 영향을 주는 것을 확인했고 entity 정보를 더 자세하고 직관적으로 전달해보기 위해 시도함
- 총 4개의 prompt를 시도하였고 없었을 때에 비해 모두 큰 성능향상을 보였으나 prompt간에 유의미한 성능 차이를 보이진 못하여 하나의 prompt를 선택하진 못함

Method	Input Example	klue/roberta-large	
		Micro F1 Score	AUPRC
default	[CLS] 부덕이는 부캠의 마스코트이다. [SEP]	47.822	38.445
s_sep_o	[CLS] **부덕이 [SEP] 마스코트** [SEP] 부덕이는 부캠의 마스코트이다. [SEP]	72.193	72.167
s_and_o	[CLS] **부덕이와 마스코트의 관계** [SEP] 부덕이는 부캠의 마스코트이다. [SEP]	**73.575**	71.588
quiz	[CLS] **다음 문장에서 부덕이와 마스코트의 관계를 추출하시오.** [SEP] 부덕이는 부캠의 마스코트이다. [SEP]	**73.299**	71.995
question	[CLS] **다음 문장에서 부덕이와 마스코트의 관계를 추출하시오.** [SEP] 부덕이는 부캠의 마스코트이다. **부덕이와 마스코트는 어떤 관계입니까?** [SEP]	**73.446**	73.399

2) Entity Representation

- 문장 내에서 entity의 위치와 형태를 직접적으로 명시해주는 entity marker를 활용해 entity간 관계를 모델이 명확하게 알 수 있도록 input을 변경함
- punct는 그렇지 않은 경우와 달리 special token(ex. [SUBJ-PER], [E1])을 추가하지 않았음을 의미
- 모든 경우에서 entity representation을 활용한 경우가 성능이 훨씬 좋았으나 이 또한 entity mask를 제외한 나머지 방법론들간에 유의미한 성능 차이를 확인할 수는 없었음

Method	Input Example	klue/roberta-large	
		Micro F1 Score	AUPRC
Default	부덕이는 부캠의 마스코트이다.	47.822	38.445
Entity Mask	[SUBJ-PER]는 부캠의 [OBJ-TITLE]이다.	68.057	66.13
Entity Marker	[E1] 부덕이 [/E1]는 부캠의 [E2] 마스코트 [/E2]이다.	72.678	**72.951**
Entity Marker (punct)	@ 부덕이 @ 는 부캠의 # 마스코트 # 이다.	72.458	72.548
Typed entity marker	<S:PERSON> 부덕이 </S:PERSON>는 부캠의 <O:TITLE> 마스코트</O:TITLE>이다.	**73.005**	72.647
Typed entity marker (punct)	@ * person * 부덕이 @ 는 부캠의 # ^ title ^ 마스코트 # 이다.	72.54	72.14

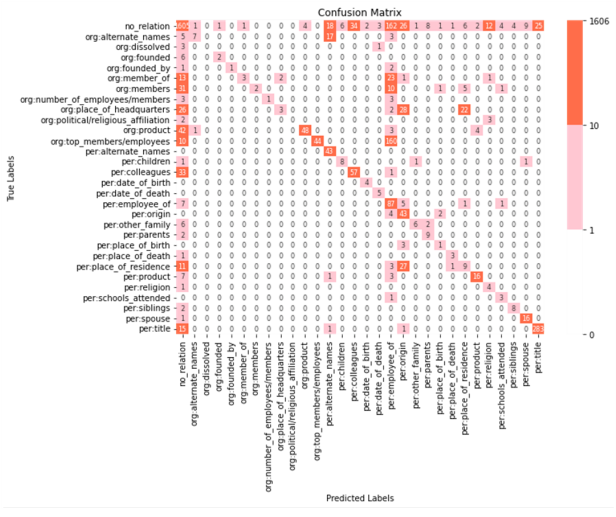
3) Prompt + Entity Representation

- prompt와 entity representaion의 여러 조합들에 대한 다각적인 실험을 진행
- 여러 조합을 확인해봤으나 하나의 조합을 설정하기엔 조합 간의 성능 차이가 미미하여 다른 요소를 추가한 실험들을 통해 결정하기로 하였음

	input_format	prompt	micro f1	auprc	loss
1	default	default	47.822	38.445	1.473
2	default	s_sep_o	72.193	**72.167**	1.021
3	default	s_and_o	**73.575**	71.588	1.032
4	default	quiz	**73.299**	71.995	1.023
5	default	problem	**73.446**	**73.399**	1.039
6	entity_mask	default	68.057	66.13	1.057
7	entity_marker	default	72.678	**72.951**	0.9561
8	typed_entity_marker	default	**73.005**	72.647	0.9917
9	typed_entity_marker_punct	default	72.54	72.14	1.082
10	typed_entity_marker_punct_한글	default			
11	entity_marker	s_sep_o	72.356	71.557	1.056
12	entity_marker	s_and_o	**72.984**	71.359	1.022
13	typed_entity_marker	s_sep_o	**72.968**	71.612	1.061
14	typed_entity_marker	s_and_o	**73.777**	71.62	1.021
15	typed_entity_marker_punct	s_sep_o	72.074	**73.906**	1.129
16	typed_entity_marker_punct	s_and_o	**72.94**	71.698	1.082

4) 학습 결과 분석

- prompt와 entity marker를 적용한 모델의 validation data에 대한 inference 결과를 confusion matrix로 분석
- 가로축이 예측한 라벨, 세로축이 실제 라벨로 대각선이 잘 예측한 결과들이고 나머지가 맞추지 못한 결과임
- no_relation 라벨과 관련된 예측이 잘 이루어지지 않고 있으며 틀린 예제를 보면 모델이 아예 엉뚱한 예측을 하고 있는 것이 아니라 관련성이 있는 라벨로 잘못 예측하는 경우가 많음
- 즉, 데이터의 전반적인 특징은 파악하고 있으나 세부적인 부분까지 엄밀하게 구분하지 못하고 있다고 판단
- 따라서 모델이 깊고 예리하게 학습할 수 있게 새로운 layer를 추가하는 방법을 고안함

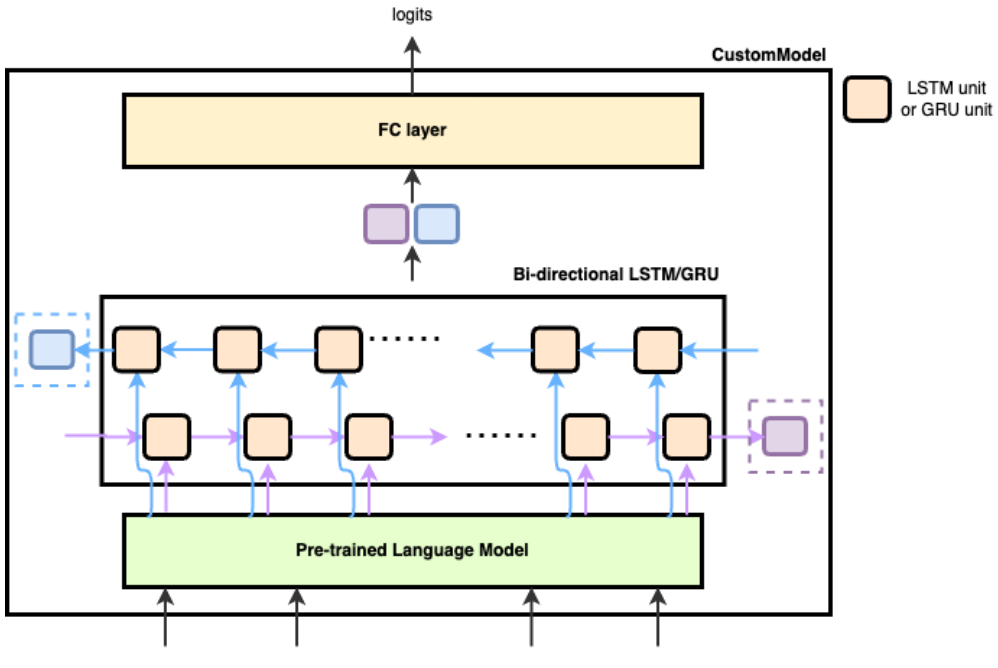


#	True Labels	Predicted Labels	value
1	per:place_of_residence	per:origin	27
2	org:place_of_headquarters	per:origin	28
3	org:place_of_headquarters	per:place_of_residence	22
4	org:alternate_names	per:alternative_names	17
5	org:member_of	per:employee_of	23
6	org:members	per:employee_of	10
7	org:top_members/employees	per:employee_of	160

실험 - Model architecture 및 Hyper parameter 변경

1) klue/roberta-large + (Bi-LSTM or Bi-GRU)

`klue/roberta-large` 모델을 기반으로 `Bi-directional LSTM`과 `GRU` layer를 각각 추가하는 두 가지 custom model을 고려하였음



1. 토큰화된 sequence를 roberta 의 입력으로 주어 각 토큰의 last hidden state를 추출
2. 해당 hidden을 bi-directional LSTM 혹은 bi-directional GRU layer에 통과시킴
3. 순방향과 역방향 두 개의 last hidden state를 concat
4. fully-connected layer를 통과시켜 최종적인 logits값을 얻게 됨

2) 모델 구조 변화 실험

- 실험군 아키텍처

  Bi-LSTM/GRU의 양방향 next hidden states

  → [Batch Norm]

  → [Activation function (GELU)]

  → [Dropout]

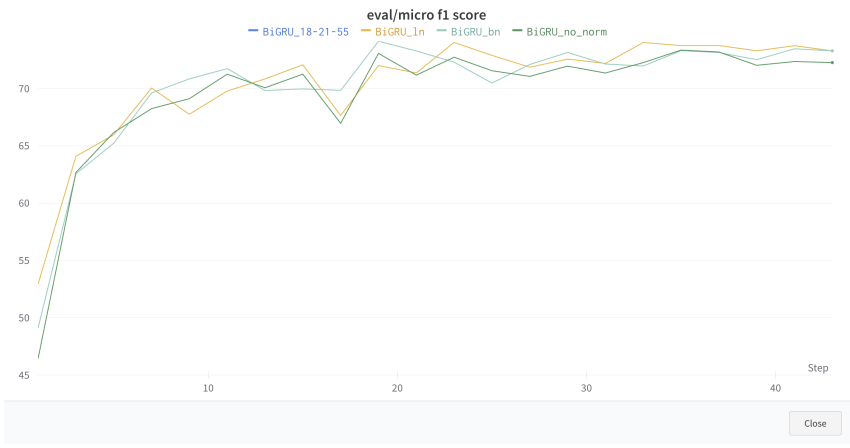
  → FC layer

- Dropout 유무, 활성화 함수 유무에 따른 성능 비교 (3가지 시드 사용, 평균치)

	기본 구조(실험군)	Dropout 제거 (대조군 1)	GELU 제거 (대조군 2)	GELU & Dropout 제거 (대조군 3)
Micro F1	72.778	73.264	71.897	**73.422**

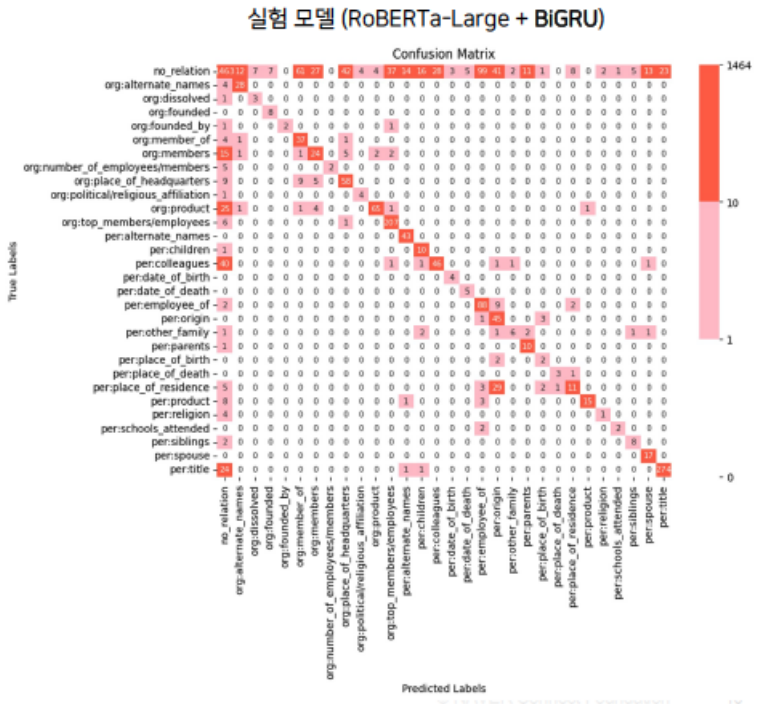
  - 결론: **GELU, Dropout을 모두 제거**하는 쪽의 기대 성능이 높을 것으로 판단되어 제거 결정.

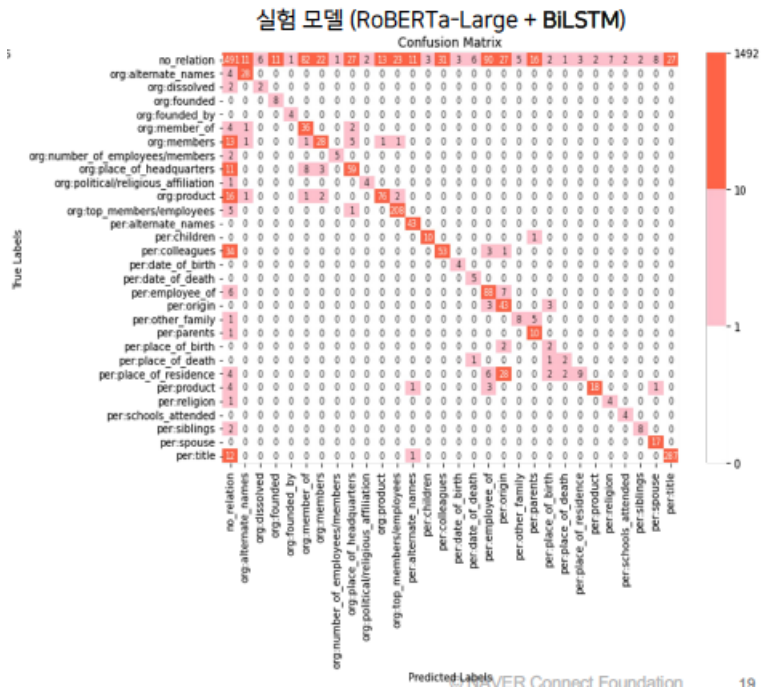
- LayerNorm, BatchNorm 추가 여부에 따른 성능 비교 분석



  - 실험 횟수가 1회이기 때문에 신뢰도가 낮으나, peak F1 기준으로 LayerNorm을 추가하였을 때 가장 좋은 성능을 보였으며, 평가지표의 수렴 구간에서도 **LayerNorm**이 안정적으로 좋은 성능을 보이는 것을 확인.

3) 학습 결과 분석





- Custom model의 validation data에 대한 inference 결과를 confusion matrix로 분석
- 기존 모델에 비해 주대각선 이외의 값들이 감소한 점에서, 모델의 예측 성능이 향상되었음을 알 수 있음
  - 실제로 기존 모델과 custom model의 결과를 비교해보면, custom model에서 False Positive와 False Negative가 모두 감소하여 `precision`, `recall`, `f1 score`가 모두 개선되었음을 확인

	기존 모델	BiGRU	BiLSTM
FP	683	583	546
FN	581	269	224
precision	0.5118	0.6381	0.6628
recall	0.5520	0.7926	0.8273
f1 score	0.5312	0.7070	0.7359

		True Class	
		no_relation	no_relation x
Predicted Class	no_relation	TP	FP ▲ 정답값은 no_relation이 아닌데 No_relation으로 예측한 값
	no_relation x	FN ▼ 정답값은 no_relation인데 다른 label로 예측한 값	TN

		True Class	
		no_relation	no_relation x
Predicted Class	no_relation	TP	FP ▼ 정답값은 no_relation이 아닌데 no_relation으로 예측한 값
	no_relation x	FN ▲ 정답값은 no_relation인데 다른 label로 예측한 값	TN

- 기존 모델에 비해 Predicted Labels = no_relation인 값들은 감소하고, True Labels = no_relation인 값들은 증가함
  - Predicted Labels = no_relation인 값의 감소는 no_relation label 기준의 False Positive가 감소했음을 의미하며, 이는 높은 precision으로 이어짐
  - True Labels = no_relation인 값의 증가는 no_relation label 기준의 False Negative가 증가했음을 의미하며, 이는 낮은 recall로 이어짐

no_relation label에 대한 recall 및 precision

	기존 모델	Bi-GRU	Bi-LSTM
precision	0.8751	**0.9020**	**0.9238**
recall	**0.8290**	0.7557	0.7701

Leaderboard 제출 결과

	기존 모델	Bi-GRU	Bi-LSTM
Micro f1	74.3558	**75.2809**	**76.5345**
auprc	76.6053	**80.2392**	**79.9067**

- 이 때, no_relation label의 precision은 다른 label에 비해 micro f1 score에 큰 영향력을 가지므로, no_relation label의 precision 지표의 개선은 유의미한 변화로 해석해볼 수 있음

4) loss function

Cross-entropy Loss		Focal Loss
$\mathrm{CE}(p_t) = -\log(p_t).$	수식	$\mathrm{FL}(p_t) = -\alpha_t(1-p_t)^\gamma \log(p_t).$
71.371	Micro F1 Score	71.225
69.653	AUPRC	**71.500**

train 데이터에 class imbalance 문제가 있음을 확인하고 cross-entropy loss 대신 focal loss를 도입함

- Cross entropy loss와 유사한 식 구조를 가지며, 정답을 맞출 확률인 p가 높은 '쉬운 문제'에 대해서는 가중치를 낮춰 loss에 미치는 영향력을 낮추어 '어려운 문제'에 보다 집중하도록 하는 방식으로 작동
- 하지만 대회 metric인 micro f1 score에 성능개선이 없고 AUPRC에서의 개선만 눈에 띔
- no_relation이 정답인 것을 per:employee_of로 예측하는 경우를 살펴보니, 주어진 sentence 정보를 넘어선 상식을 필요로 하는 경우가 있는 등 '쉬운 예제'와 '어려운 예제' 간의 경계가 모호한 경우가 있어서 focal loss가 효과적이지 못했던 것이라고 판단함

지난 5일 아이즈원 해외 팬은 자신이 강혜원 씨 이름으로 스코틀랜드에 있는 땅을 샀다는 글과 함께 인증 사진을 게시했다.	{'word': '강혜원', 'start_idx': 21, 'end_idx': 23, 'type': 'PER'}	{'word': '아이즈원', 'start_idx': 6, 'end_idx': 9, 'type': 'POH'}	per:employee_of	wikitree

하이퍼파라미터 튜닝 및 최종 모델

- WandB Sweep을 활용한 Bayesian search로 파라미터 튜닝 진행
- 모델을 Base, Bi-LSTM, Bi-GRU 3개를 각각 tuning 진행 후 성능 비교
- 최종적으로 성능이 높은 5개 output을 hard voting
- 하이퍼파라미터 튜닝 상세내역

    - input_format : [default, entity_marker, entity_marker_punct, typed_entity_marker, typed_entity_marker_punct]
    - prompt : [default, s_sep_o, s_and_o, quiz, problem]
    - type_transform : [true, false]
    - lr : [1e-5, 2e-5, 3e-5, 5e-5]
    - epochs : [3, 5]
    - batch size : 64
    - adam_beta2 : [0.98, 0.999]
    - warmup_ratio : [0.06, 0.1]

    - RoBERTa 논문에서 '사전 학습' 시 사용했다고 한 AdamW의 $\beta_2 = 0.98$을 파인 튜닝 시에도 사용해 실험 진행
    - RoBERTa 논문에 따라 learning rate scheduling 시 warmup ratio = 0.06을 사용

3. 프로젝트 고찰

잘한 점 및 아쉬운 점

프로젝트 시작 전 설정한 목표의 달성 여부

구분	내용	달성 여부
실험	노션을 통한 체계적인 실험 관리: [가설 수립 → 실험을 통한 가설 검증 → 사후 분석]의 프로세스를 노션에 기록	⭕
	성능 향상을 위한 실험과 더불어 새로운 아이디어를 기반으로 하는 실험 (decoder로 encoder 따라잡기, TAPT, entity embedding)	🔺
	논리와 근거를 바탕으로 하는 실험 전개	⭕
	같은 조건의 실험이라도 seed를 바꿔가며 여러 번 진행한 후 평균을 계산하여 실험 결과의 신뢰성 높이기	🔺
	실험 이후 적극적인 사후 분석	⭕
	nohup을 통한 실험 자동화	⭕
GitHub	Git branching 전략 활용	⭕
	issue 및 Pull Request를 통해 각자 진행하고 있는 task를 파악하고, 앞으로의 계획 수립	⭕
	Commit message / branch name / issue template / PR template convention 합의 후 그에 맞춰 작성	⭕
협업	프로젝트 시작 전 전체적인 timeline을 구상함으로써 시간을 효율적으로 사용	⭕
	PM 제도의 도입으로 원활한 회의 및 프로젝트 진행	⭕
	Kanban board를 활용하여 각자 어떤 task를 진행중인지 빠르게 확인	⭕

아쉬운 점

- 모든 실험들에 대한 사후분석을 진행하지는 못했다. → 기록과 분석의 중요성..
- LSTM을 추가함으로써 성능이 어떤식으로 좋아지게 된 것인지는 알아냈지만, LSTM의 어떠한 특징 때문에 그러한 변화가 일어났는지는 알아내지 못했다.

  → architecture에 변화를 줄 때에는 해당 task의 SOTA 모델의 architecture를 참고하자
- Decoder 계열의 모델로 encoder 계열의 모델의 성능을 따라잡아보려는 시도를 계획했으나 실험까지 이어지지는 못했다.
- TAPT 및 entity embedding 실험을 진행해보았지만, 그 과정에서 발생한 문제들을 완전히 해결하지는 못했다.

다음 프로젝트에서 도전할 것

- ChatGPT와 대화하는 스페셜 미션을 즐겁게 수행하기
- Debugger를 적극적으로 활용하기
- 실험 전 문제에 대한 사전 조사를 통해 보다 다양한 논문을 소화하고 프로젝트에 반영하기
- 코드 리뷰 철저히 하기
- 깃 저장소에서 더 완성된 컨벤션 활용하기
- 가능하다면 여러 대의 GPU를 동시에 효율적으로 활용하는 방법 적용해보기