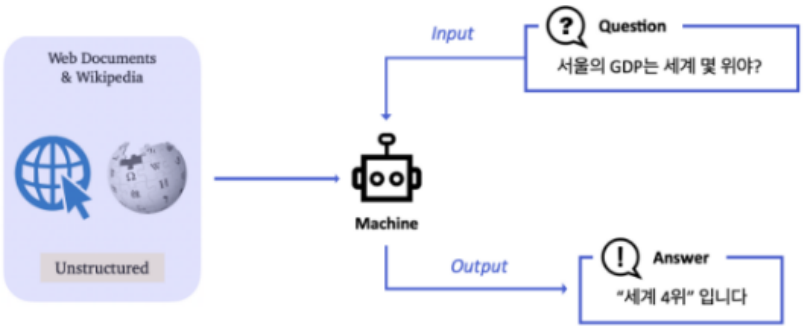
MRC WrapUp Report

프로젝트 개요

Linking MRC and Retrieval: Open-domain Question Answering (ODQA)

ODQA: 지문이 따로 주어지지 않음. 방대한 World Knowledge에 기반해서 질의응답



Open-Domain Question Answering (ODQA) 은 주어지는 지문이 따로 존재하지 않고 사전에 구축되어 있는 Knowledge resource 에서 질문에 대답할 수 있는 문서를 찾는 과정입니다. ODQA는 two-stage로 질문에 관련된 문서를 찾아주는 **"retriever"**, 관련된 문서를 읽고 적절한 답변을 찾아주는 **"reader"** 로 구성되어 있습니다. 두 가지 단계를 각각 구성하고 통합하여 어려운 질문을 던져도 답변을 해주는 ODQA 시스템을 만들고자 하였습니다.

프로젝트 팀 구성 및 역할

👩‍💻김한성: Data team // EDA + BaseLine update + Poly encoder

👩‍💻염성현: Project Manager // EDA + Data Augmentation & Pre-process + Dense Retriever

👩‍💻이재욱: Model team // Experiment Support + Ensemble

👩‍💻최동민: Data team // EDA + Experiment Support

👩‍💻홍인희: Model team // BaseLine update + BM25 + Post-process

프로젝트 수행 절차 및 방법

데이터 분석 및 전처리, 증강

EDA

수행 내용

- 중복 데이터 및 결측 데이터 분석
- 질문과 답변 쌍 분석

- 적합하지 않은 질문 분석
- Title, context, question, answer, tokenized context 등의 특성 및 분포 분석
- Answer start index, end index 분석

중복 데이터 및 결측 데이터 확인

- 중복 데이터 확인, 동일 context의 데이터가 다수 존재하였음, 질문은 상이하였음

duplicate data

```
[10]: train_df['context'].describe()

[10]: count                                                 3952
      unique                                                3340
      top       그러나 한편으로는 미국 사회의 인종차별주의적, 흑인을 차별하는 태도를 목격하면서 백...
      freq                                                     4
      Name: context, dtype: object

[17]: train_df[train_df['context'].duplicated()].sort_values(by='context_len')
```

- 결측 데이터는 존재하지 않았음

```
train_dataframe.isnull().sum()
valid_dataframe.isnull().sum()
test_dataframe.isnull().sum()
```

✓ 0.3s

```
title              0
context            0
question           0
id                 0
answers            0
document_id        0
__index_level_0__  0
dtype: int64
```

질문과 답변 쌍 분석

- 하나의 질문에 여러 개의 답변이 존재하는 경우는 없었음

```
answer_cnt = 0
for i, df in train_dataframe.iterrows():
  if len(df['answers']['text'])!=1:
    answer_cnt += 1
    print(df['answers']['text'])
print(answer_cnt)
```

✓ 0.2s Python

```
0
```

적합하지 않은 질문 분석

- 지시어나 인칭대명사는 ODQA에 적절하지 않다고 파악
- '그'와 '그것'이라는 단어를 포함한 질문 조사

```
for i, df in train_dataframe.iterrows():
    if "그는" in df['question']:
        print(df['question'])
```

0.1s

```
도스토옙스키의 시력이 망가졌을 때 그는 어떤 방식으로 작품을 완성할 수 있었나?
유일한 생존자가 정신을 차렸을 때 그는 누가 유괴당했다는 것을 알게되는가?
```

```
for i, df in train_dataframe.iterrows():
    if "그가" in df['question']:
        print(df['question'])
```

0.1s

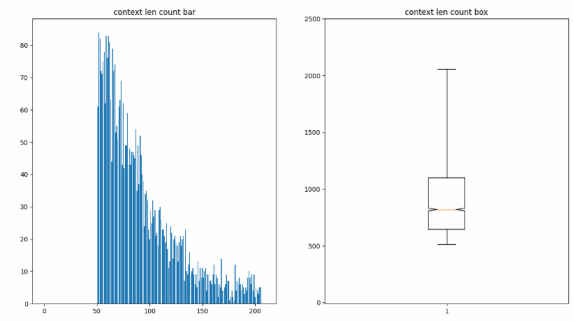
```
돈 칭자오를 때렸으며, 그가 원한을 품게 한 인물은?
래드왈드 왕이 죽고난 뒤 그가 다스리던 지역은 어떤 왕국이 통치하게 되었나?
마루게*의 끈기에 감탄하여 그가 학교 다니는 것을 허락했던 인물은?
보그가 만든 이메일 서비스는 무엇을 대상으로 만들어졌나?
위그가 라울로부터 물려받은 직위는 무엇인가요?
하산 부즈루그가 쿠첵에게 패배 후 쫓겨난 곳은?
하산 쿠첵과의 싸움에서 진 하산 부즈루그가 결국 쫓겨간 곳은?
```

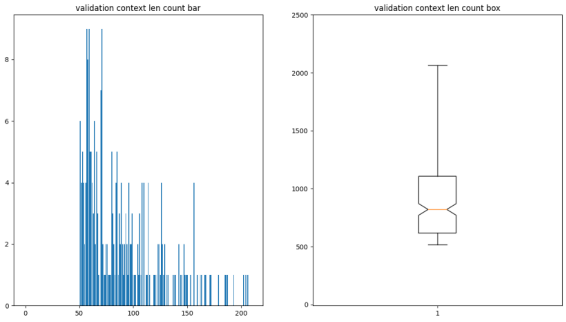
→ 고유명사가 아닌 단어가 포함된 질문은 ODQA에 적합하지 않다고 예상할 수 있었으나 질문 판별 및 정제는 따로 진행하지 않았음. 판별 및 정제가 임의적이게 될 것을 우려

Title 특성 분석

- train : 2716개의 타이틀 (총 3952개의 데이터 중)
- validation : 228개의 타이틀 (총 240개의 데이터 중)
- wiki : 31755 개의 타이틀 (총 60613 개의 데이터 중)

Context 분포 분석



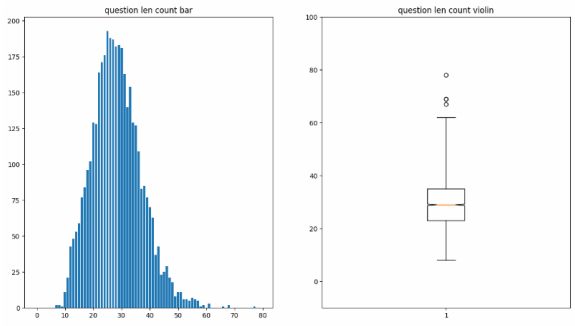


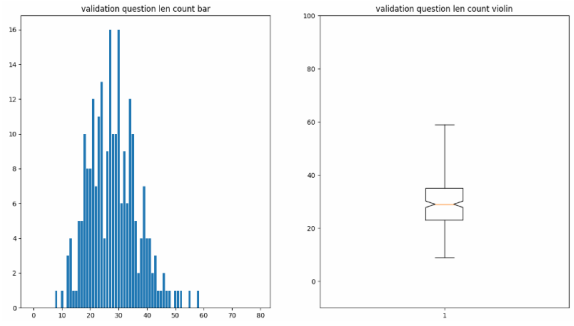
- train data
  - 최소 길이 : 512
- validation data
  - 최소 길이 : 517

- 평균 : 920
- 최대 길이 : 2059

- 평균 : 916
- 최대 길이 : 2064

Question 분포 분석

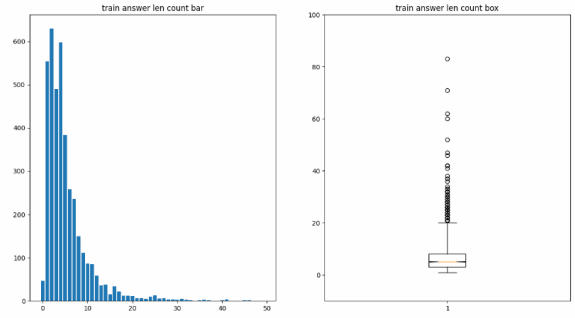


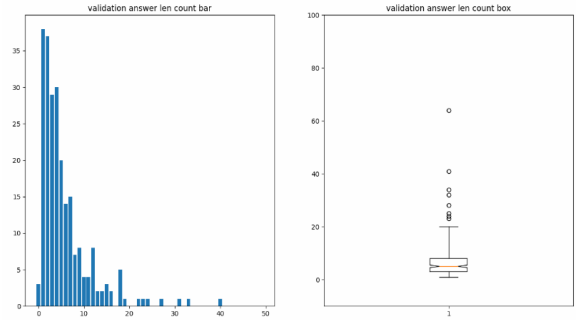


- train data
  - 최소 길이 : 8
  - 평균 길이 : 29
  - 최대 길이 : 78
- validation data
  - 최소 길이 : 9
  - 평균 길이 : 29
  - 최대 길이 : 59

→ Context와 Question의 최소 길이 및 최대 길이는 Augementation 데이터 정제에 활용

Answer 분포 분석

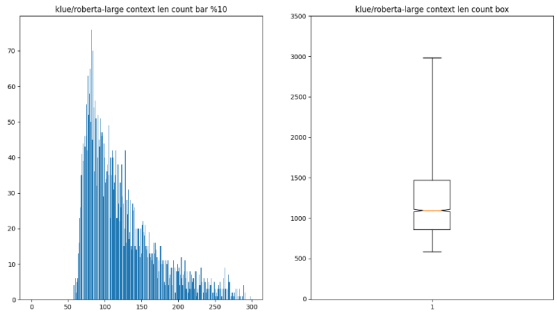


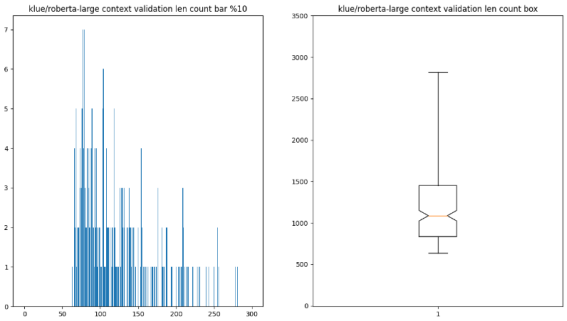


- train data
  - 최소 길이 : 1
  - 평균 길이 : 6.2
  - 최대 길이 : 83
- validation data
  - 최소 길이 : 1
  - 평균 길이 : 6.9
  - 최대 길이 : 64

→ Answer의 길이가 30이 넘는 데이터가 25개에 불과함을 발견, max_answer_length를 30으로 유지

Tokenized context 분포 분석





- train data
  - 최소 길이 : 585
  - 평균 길이 : 1227
  - 최대 길이 : 2983
- validation data
  - 최소 길이 : 634
  - 평균 길이 : 1222
  - 최대 길이 : 2818

→ tokenize 이후 각각의 sequence 길이가 대체로 상승하였음을 발견, special token과 #이나 _ 등의 구분자의 역할이 컸음.

Tokenized sequence 분석

[11]:
```
from transformers import AutoTokenizer
# 어떤 토크나이저에 따라 길이는 달라집니다. 항상
tokenizer1 = AutoTokenizer.from_pretrained('klue/bert-base')
tokenizer2 = AutoTokenizer.from_pretrained('monologg/kobigbird-bert-base')
```

[20]:
```
train_max = 0
for i,df in train_df.iterrows():
    tmp = len(tokenizer2(df['context'])['input_ids'])
    tmp2 = len(tokenizer2(df['question'])['input_ids'])
    train_max = max(train_max,tmp+tmp2)
train_max
```

[20]: 1180

[19]:
```
val_max = 0
for i,df in val_df.iterrows():
    tmp = len(tokenizer2(df['context'])['input_ids'])
    tmp2 = len(tokenizer2(df['question'])['input_ids'])
    val_max = max(val_max,tmp+tmp2)
val_max
```

[19]: 1165

→ input_ids의 길이를 각각 1180, 1165임을 확인하고 kobigbird 의 max_seq_len을 1200으로 설정

Answer start index, end index 정답 일치 여부 확인

- answer_start와 [answers]['text'] 모두 일치

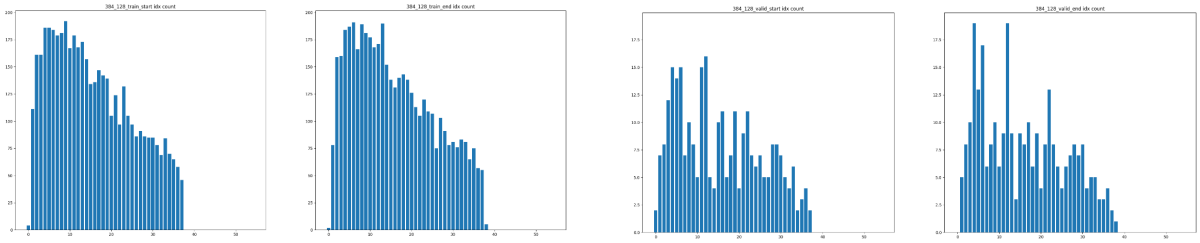
```python
answer_match_cnt = 0
temp_cnt = 0
for i, df in train_dataframe.iterrows():
    # print(df['answers']['text'])
    # print(df['context'][int(*df['answers']['answer_start'])])
    if df['answers']['text'][0] != df['context'][int(*df['answers']['answer_start']):int(*df['answers']['answer_start'])+len(df['answers']['text'][0])]:
        print('wrong answer')
        print(df['answers']['text'][0])
        print(df['context'][int(*df['answers']['answer_start']):int(*df['answers']['answer_start'])+len(df['answers']['text'][0])])
        answer_match_cnt += 1
    temp_cnt += 1
print(answer_match_cnt)
print(temp_cnt)
```

✓ 0.2s

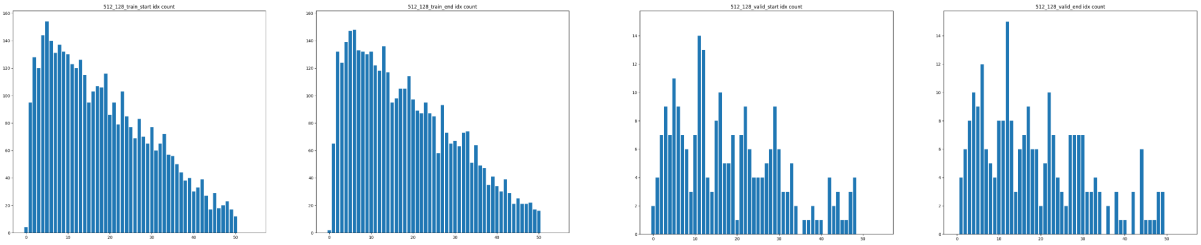
```
0
3952
```

Answer start index, end index 분포 분석

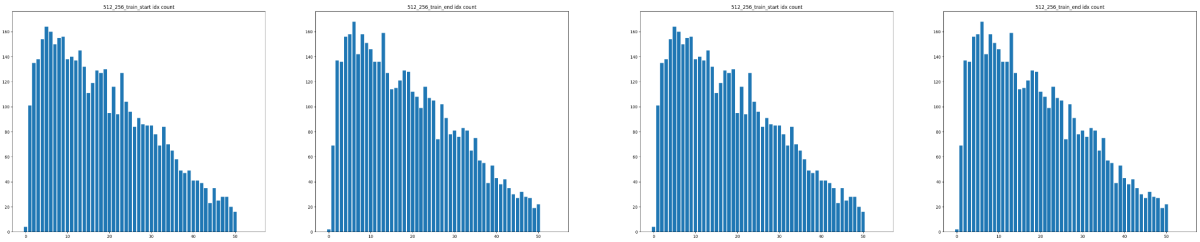
- max length = 384, stride = 128의 Train, validation distribution



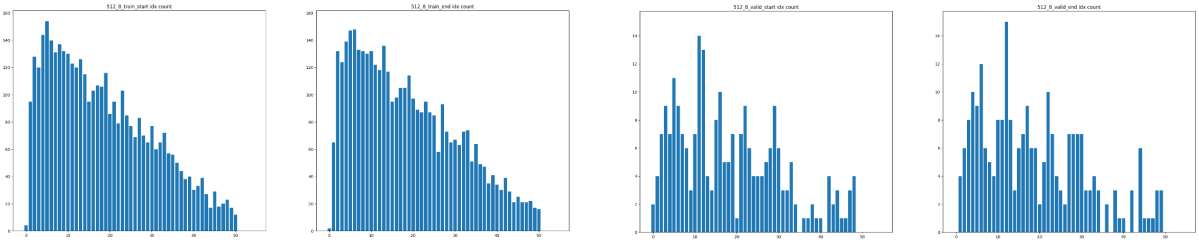
- max length = 512, stride = 128의 Train, validation distribution



- max length = 512, stride = 256의 Train, validation distribution



- max length = 512, stride = 8의 Train, validation distribution



→ Max Length가 분포의 형태에 영향을 준다는 것을 발견

→ Stride가 Sequence의 개수 및 정답 위치 찾는 것에 영향을 준다는 것을 발견

실험 결과

Max Length

Aa Model	# Max Le...	# L_EM	# L_F1	# V_EM	# V_F1	Tot_Hour
monologg/kobigbird-bert-base	1200	37.08	47.79	60	69.9241	0:48:35
klue/bert-base OPEN	512	31.67	44.77	55.83	63.6458	0:35:03
klue/roberta-large	512	40.42	53.59	65	72.744	
xlm-roberta-large	514	27.08	41.92	51.25	58.7259	1:59:14.58
monologg/kobigbird-bert-base	384	36.25	47.85	60.4167	69.2986	
klue/bert-base	384	35.42	47.3	54.17	64.01	
klue/roberta-large	384	34.58	50.01	62.9167	72.0369	
xlm-roberta-large	384	29.58	43.56	50.41	60.84	

Stride

Aa 이름	# epoch	# max_len	# doc_stri...	# L_EM	# L_F1	# V_EM	# V_F1
klue/roberta-large	10	512	256	41.67	53.19	66.667	74.048
monologg/kobigbird-bert-base	10	512	256	34.58	47.52	55.8333	65.633
klue/bert-base	10	512	256	33.33	45.89	53.3333	61.1708
xlm-roberta-large	10	512	256	30.83	47.04	47.5	56.7813
klue/roberta-large	10	512	8	35	43.82	55	62.343
monologg/kobigbird-bert-base	10	512	8	33.75	45.74	55.83	66.9907
klue/bert-base	10	512	8	31.25	42.2	52.9167	62.1636
xlm-roberta-large	10	512	8	25	37.61	40	50.7165
klue/roberta-large	10	512	128	40.42	53.59	65	72.744
monologg/kobigbird-bert-base	10	1200	128	37.08	47.79	60	69.9241
klue/bert-base	10	512	128	31.67	44.77	55.83	63.6458
xlm-roberta-large	10	514	128	27.08	41.92	51.25	58.7259
monologg/kobigbird-bert-base	10	512	128	35	47	61.6667	70.3548

- Klue/roberta-large 사용 시 Max Length 512, Stride 256가 최적이었음을 확인

Augmentation

KLUE MRC

3.7.1 Dataset Construction

Source Corpora First, we collect passages from Korean WIKIPEDIA and news articles provided by The Korea Economy Daily and ACROFAN. WIKIPEDIA articles are one of the most commonly used resources for creating MRC datasets. We additionally include news articles reporting contemporary social issues to enhance diversity of passages. They are provided by The Korea Economy Daily and ACROFAN. As news articles are generally copyrighted work, we sign a contract with the news providers to use and redistribute the articles under CC BY-SA license only for building a dataset for machine learning purposes. We believe multi-domain corpus can help MRC models enhance their generalizability.

We preprocess the corpus to collect passages. For WIKIPEDIA articles, we remove duplicates in other existing Korean MRC benchmarks (e.g., KorQuAD) for precise evaluation of models. Then, we split each article by its sections to obtain passages. For the news articles, we filter out political articles and articles belonging to categories which have less than 100 articles. We finally gather all preprocessed passages whose length is longer than 512 and shorter than 2048 in characters.

- WIKIPEDIA와 뉴스 데이터를 출처로 활용
- KorQuAD와 겹치는 passage는 MRC 데이터셋에서 제거하였음을 확인

→ KorQuAD 데이터 증강, 뉴스 기반 데이터셋 증강 아이디어 착안

Dataset Search

이름	설명
KorQuAD v1.0	Wikipedia article 문단 내에서 답을 찾는 데이터셋
KorQuAD v2.1	Wikipedia article 전체에서 답을 찾는 데이터셋
AI HUB 기계 독해	기계독해 개발에 활용될 수 있는 뉴스 본문 기반 학습 데이터셋

- 증강에 사용할 후보 데이터셋을 찾아보고 특성을 비교하였음
- KorQuAD v2.1의 경우, 표와 리스트 등을 포함하고 context의 길이 또한 길어 각각을 전처리하고 train data의 최대 길이를 넘지 않는 데이터셋만 사용하기로 하였음

전처리

- KorQuAD v2.1 데이터 기반으로 전처리 함수 작성
- json 태그를 활용해 테이블 및 리스트 제거
- 끝맺지 않은 문장을 찾아 처리하는 방식으로 문서 부연을 제거
- 1만 여개 이상의 데이터를 직접 보고 전처리
- random 함수를 활용해 전처리 성능을 확인하고 추가 처리

증강 결과

```
DatasetDict({
    train: Dataset({
        features: ['title', 'context', 'question', 'id', 'answers', 'document_id', '__index_level_0__'],
        num_rows: 209972
    })
    validation: Dataset({
        features: ['title', 'context', 'question', 'id', 'answers', 'document_id', '__index_level_0__'],
        num_rows: 22280
    })
})
```

- train data 209972 개, validation data 22280 개의 전체 dataset 구축

실험 결과

Augmentation

Aa 이름	augmen...	Retriever	# epoch	# batch_si...	# L_EM	# L_F1	# V_EM	# V_F1
SOTA: klue/roberta-large (OPEN)	aihub_10%, korquad_2.1, korquad_1.0	BM25	10	256	62.5	75.65	77.17	85.61
klue/roberta-large	korquad_2.1, korquad_1.0, aihub	BM25	15	8	62.5	74.34	71.409	85.542
SOTA : klue/roberta-large	korquad_1.0	BM25	15	8	61.25	73.21	87.529	92.272
SOTA: klue/roberta-large	korquad_1.0, aihub_10%	BM25	15	8	62.5	74.39	82.5	90.14
monologg/koelectra-base-v3-discriminator	korquad_2.1, korquad_1.0, aihub	BM25	10	8	49.58	64.28	78.96	84.57

- 증강 데이터 전체를 썼을 때 성능이 더 우수해졌음을 확인

BaseLine Update

업데이트 전

- nested 구조로 인한 프로젝트 흐름 파악 불용이
- 에러 발생 지점 트래킹에 어려움 존재

업데이트 후

- train.py 코드 감소 420 줄 → 129 줄
- nested 구조의 문제점 해결
  - 추가로 MRC_Dataset을 만들어 WandB에서 eval_loss가 확인 가능해짐
- task별 작업을 분류
  - 실험 관리가 원활해짐
- OmegaConf를 통해 기존의 복잡한 parser를 ‘config’ parser로 단순화

```
baseline/
│
├── train.py
├── inference.py
├── arguments.py
├── main.py
│
├── config/ - abstract all config for model
│   ├── config.yaml
│   └── sweep_config.yaml
│
├── model/
│   ├── reader.py
│   └── retrieval.py
│
├── trainer/
│   └── trainer.py  -> trian_qa.py 변형 및 통합
│
├── utils/
│   ├── load_data.py
│   ├── util_qa.py
│   └── util.py
│
│
└── thanks for comming I'm Yeombora
```

Reader 모델

Parameter Search

기능 별 Ablation study 진행

PLM

Aa model_name	# Leader ...	# Leader_F1	EM / F1	Person	MaxSeq...	epoch / ...	# Paramet...	총 학습 ..
monologg/kobigbird-bert-base	36.25	47.85	60.4167 / 69.2986	한성 김	384	10 / 8	117,888,770	00:41:24
Seongmi/kobigbird-bert-base-finetuned-klue-v2_epoch64	35.83	46.85	68.3333 / 76.8941	Sunghyun Youm	384	10 / 8	117,888,770	00:41:44
klue/bert-base	35.42	47.3	54.17 / 64.01	이니	384	10 / 8	110,028,290	00:35:36
monologg/koelectra-base-v3-finetuned-korquad	35	45.56	59.16667/67.144	한성 김	384	10 / 8	112,332,290	00:46:21
klue/roberta-base	34.58	46.62	60.4167 / 68.2673	동민 최	384	10 / 8	110,029,058	0:35:08.75
klue/roberta-large	34.58	50.01	62.9167 / 72.0369	동민 최	384	10 / 8	335,608,834	1:48:01.16
monologg/koelectra-base-v3-discriminator	32.92	41.75	56.25 / 64.1796	재욱 이	384	10 / 8	112,332,290	0:36:18
xlm-roberta-large	29.58	43.56	50.41 / 60.84	Sunghyun Youm	384	10 / 8	558,842,882	02:07:28
monologg/koelectra-small-v3-discriminator	24.58	34.51	42.5 / 50.1224	재욱 이	384	10 / 8	14,056,706	0:12:34
xlm-roberta-base	22.92	35.86	45.00 / 52.00	Sunghyun Youm	384	10 / 8	277,454,594	00:43:13
klue/roberta-small	30.42	42.64	49.5833 / 57.8464	이니	384	10 / 8	67,501,826	00:18:58

- 각 PLM 모델의 성능 확인 (위는 PLM 실험 예시)

BatchSize

이름	optim	Scheduler	epoch	batch_si...	max_len	stride	L_EM	L_F1	V_EM	V_F1
klue/roberta-large	SGD	CyclicLR	30	8	512	256	42.92	54.49	74.1667	83.2402
klue/roberta-large	SGD	CyclicLR	20	16	512	256	40.42	53.14	74.5833	81.9551
klue/roberta-large	SGD	CyclicLR	20	16	384	128	40.42	53.14	73.3333	81.5105
klue/roberta-large	SGD	CyclicLR	20	32	384	128	38.75	51.86	72.9166	81.0106

- SOTA 모델인 klue/roberta-large의 최적 Epoch, Batchsize 확인

각 기능 별 Ablation study를 통합하여 SOTA 모델 표 작성

Experiments

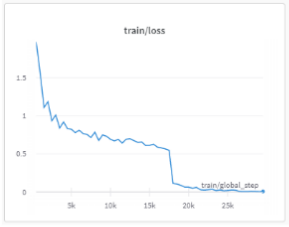
Description	Model_name	Data_Aug	Retriever	L_EM	L_F1	Rando...	epoch	batch_si...	max_len	stride	scheduler	ensemble
SOTA Ablation	klue/roberta-large		TF-IDF	42.92	54.49	42	30	8	512	256	SGD / CyclicalLR	
BaseLine	monologg/kobigbird-bert-base		TF-IDF	36.25	47.85	42	10	8	384	128	adamW / LambdaLR	
PLM Search	monologg/kobigbird-bert-base		TF-IDF	36.25	47.85	42	10	8	384	128	adamW / LambdaLR	
Max Length	klue/roberta-large		TF-IDF	40.42	53.59	42	10	8	512	256	adamW / LambdaLR	
Epoch Plus	klue/roberta-base		TF-IDF	36.67	46.48	42	30	8	384	128	adamW / LambdaLR	
Stride_Search	klue/roberta-large		TF-IDF	41.67	53.19	42	10	8	512	256	adamW / LambdaLR	
Optim Difference	klue/roberta-large		TF-IDF	42.92	54.49	42	30	8	512	256	SGD / CyclicalLR	
BatchSize Search	klue/roberta-large		TF-IDF	42.92	54.49	42	30	8	512	256	SGD / CyclicalLR	
Augmentation	klue/roberta-large	Korquad_v1.0 Korquad_v2.1 AI_Hub_10%	BM25	62.5	75.65	42	10	256	512	256	SGD / CyclicalLR	
DPR_BM25	klue/roberta-large		BM25	55	68.05	42	15	8	512	256	SGD / CyclicalLR	
Ensemble	klue/roberta-large	Korquad_v1.0 Korquad_v2.1 AI_Hub_mindslab AI_Hub_10%	BM25	68.33	78.24	42					SGD / CyclicalLR	Soft-Voting

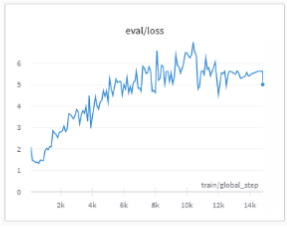
학습 설정

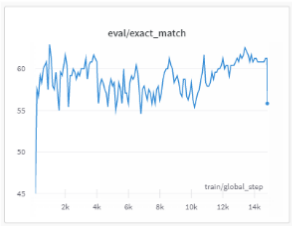
Scheduler & Optimizer

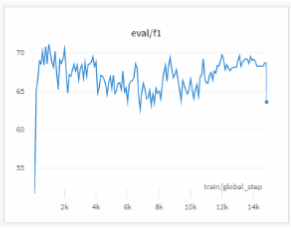
Overfitting에 대한 염려 : 학습 진행에 따라 evaluation loss가 지속 상승

`Lambda Learning Rate with AdamW (default)`

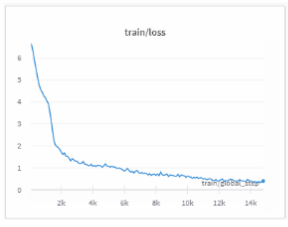


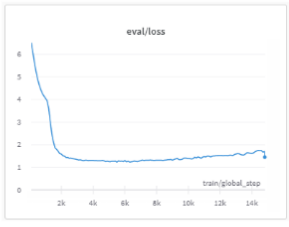


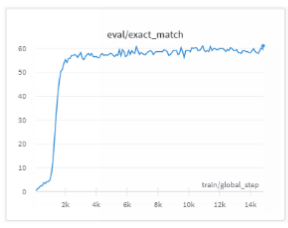


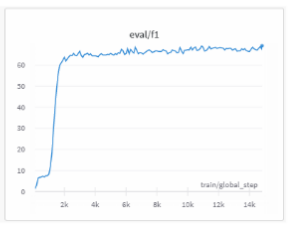


`Cyclical Learning Rate with SGD`









`Cyclical Learning Rate` 를 사용한 경우, evaluation loss가 꾸준히 내려가는 결과를 관측하였음

실험 결과

Optimizer

이름	optim	Scheduler	epoch	max_len	stride	L_EM	L_F1	V_EM	V_F1
klue/roberta-large	SGD	CyclicLR	30	512	256	42.92	54.49	74.1667	83.2402
klue/roberta-large	AdamW	CosineAnnealingWarmRestarts	30	512	256	41.67	53.55	71.25	78.866
klue/roberta-large	SGD	CyclicLR	10	512	256	41.67	53.55	72.5	80.3861
klue/roberta-large	AdamW	LambdaLR	10	512	256	41.67	53.19	66.667	74.048
monologg/kobigbird-bert-base	SGD	CyclicLR	30	1200	128	30.83	44.6	61.25	69.25

- SGD - CyclicLR을 사용했을 시에 validation set에 대한 학습이 더 잘이루어졌음을 확인
- Scheduler와 Optimizer 설정이 학습이 general한 방향으로 이루어지도록 도움을 주었다고 판단

Retriever 모델

- 질문에 관련된 문서를 찾아주는 “retriever” 개선으로 모델 성능 향상을 꾀함

BM25

- TF와 IDF를 곱해준다는 시스템은 동일, 보정 파라미터들을 추가해서 성능을 개선한 알고리즘

$$score(D,Q) = \sum_{i=1}^{n} IDF(q_i) \cdot \frac{f(q_i, D) \cdot (k_1 + 1)}{f(q_i, D) + k_1 \cdot (1 - b + b \cdot \frac{|D|}{avgdl}}$$

$avgdl$: 평균 문서의 길이

k_1 : smoothing parameter (default : 1.2)

b : default : 0.75

질문에 관련된 문서 일치 여부 (총 4192 개)

Top - 1	TF-IDF	BM25
일치	1056 개	2260 개
불일치	3136 개	1932 개

Top-K 실험

Top - k	correct retrieval
k = 1	0.3406
k = 20	0.9067
k = 50	0.9384

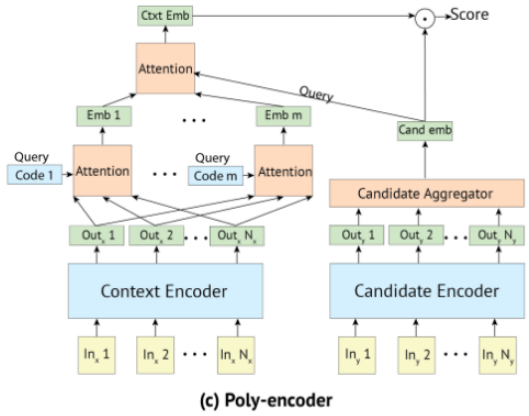
실험 결과

DPR_BM25

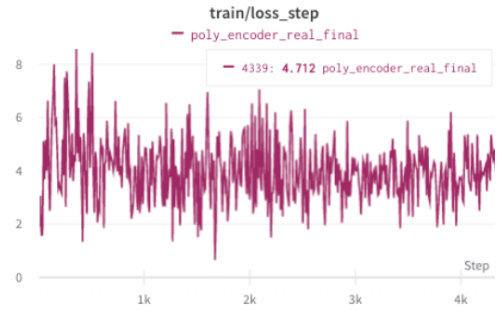
이름	DPR	BM25	KorQuAD	L_EM	L_F1	V_EM	V_F1
klue/roberta-large	X	X	X	43.33	53.67	69.5833	78.6956
klue/roberta-large	X	O	X	55.00	68.05	69.5833	78.6956

- BM25 적용 후, LeaderBoard의 EM, F1이 크게 향상되었음을 확인

Poly Encoder



(c) Poly-encoder



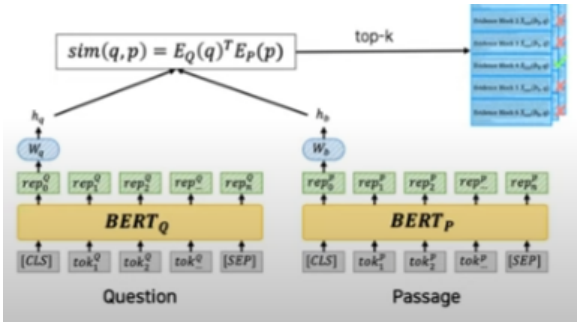
모델 구조 및 구성

- dot-production ⇒ Bi-encoder + Learnable Parameter code
- Passage vector와 Attention 연산을 통해 query vector를 생성 시 Passage에 영향을 받도록 학습
- 질문이 문서와 관련 있을 경우,
  - 임베딩 벡터상 유사하게 표현
- 질문이 문서와 관련이 없을 경우,
  - 임베딩 벡터상 L2거리를 멀리 함

실험 결과 및 한계

- 4,000 step에서 loss가 감소하는 추세를 보이지 않고 발산하는 과정이 보였음
- Optimizer, Scheduler에 대한 고려 부족

Dense Retriever



실험 한계

- Backbone 모델을 변경하고 Batch size를 늘려보는 등 실험을 하였으나
  모델 성능을 높이지 못하였음

추가 과제

- Encoder 학습 목표 설정
- Faiss 인자 분석 및 Tuning

Ensemble

방법론

- $Blend = Soft + 0.01Hard$

수식

- **Soft**
  - 리더보드 EM, predictions probability 활용
  - **EM** = ($\frac{EM_1}{sum(EM)}$, $\frac{EM_2}{sum(EM)}$,…, $\frac{EM_x}{sum(EM)}$)
  - $soft = \sum_{i=1}^{n} prob_i EM_i$
- **Hard**
  - $Hard$ =count($TEXT_\epsilon prob$)

soft voting

11	Finished	63.3300	74.0900

Blend

15	Finished	67.5000	77.9300

참고 문헌

Big Bird: Transformers for Longer Sequences (2021)
https://arxiv.org/pdf/2007.14062.pdf

Cyclical Learning Rates for Training Neural Networks (Smith, 2017)
https://arxiv.org/abs/1506.01186

Dense Passage Retrieval for Open-Domain Question Answering (2020)
https://arxiv.org/pdf/2004.04906.pdf

Latent Retrieval for Weakly Supervised Open Domain Question Answering (Kenton Lee, 2019) https://arxiv.org/pdf/1906.00300.pdf

Poly-encoders: Transformer Architectures and Pre-training Strategies for Fast and Accurate Multi-sentence Scoring (2019)
https://arxiv.org/pdf/1905.01969.pdf

KorQuAD (URL) https://korquad.github.io/

KLUE-MRC (URL) https://klue-benchmark.com/

개인 회고

김한성

이번 프로젝트에서 나의 목표는 무엇이었는가?

- ODQA관련 논문 스터디에 참여하게 되면서 최신 논문들을 본 대회에 적용해보는 것이 목적
- 베이스라인 코드가 다소 복잡했기에 오류 수정을 하는 것이 목적

나는 내 학습 목표를 달성하기 위해 무엇을 어떻게 했는가?

- 논문 스터디 중 적용가능한 것을 모색 및 구현
- 베이스라인 코드 수정을 위해 리팩토링 관련 자료 조사 및 적용

나는 어떤 방식으로 모델을 개선하였는가?

- Passage의 특성인 긴 문장을 다루는데 있어 Answer passage가 있을 수 있도록 적용하는 것이 목적

내가 한 행동의 결과로 어떤 지점을 달성하고, 어떠한 깨달음을 얻었는가?

- 성능을 높이기 위해서는 다양한 관점이 필요하다. → 모델 적용 관점이 아니라 라벨의 특성(추출이 좋은지 생성이 좋은지 등)

전과 비교해서, 내가 새롭게 시도한 변화는 무엇이고, 어떤 효과가 있었는가?

- 끝나고 혼자 계속 생각하기, 말하면서 정리해보기
- 이전 코드를 보면서 적용하는 것 보다 효율적인 시간 관리가 가능했다.

마주한 한계는 무엇이며, 아쉬웠던 점은 무엇인가?

- 논문스터디로 구현해본 결과 명확한 성능 분석을 하기 애매했다. → 본 대회가 answer span이 명확히 존재하는 것이 성능이 더 좋기 때문

한계/교훈을 바탕으로 스스로 새롭게 시도해볼 것은 무엇일까?

- 나만의 코드 템플릿을 정해두자. → 이를 활용해 어떤 태스크가 와도 간단히 적용가능하게 발빠르게 대회를 진행할 수 있게 하자.

이재욱

이번 프로젝트에서 나의 목표는 무엇이었는가?

- BaseLine Code 이해하고 논문 구현을 통한 성능 향상이 목표였다.
- Ensemble을 할 때, 가중치를 부여하여 좋은 성적이 목표였다.

나는 내 학습 목표를 달성하기 위해 무엇을 어떻게 했는가?

- 논문에 대한 인사이트를 얻기 위해 논문 스터디에 참여하였다.

나는 어떤 방식으로 모델을 개선하였는가?

- 여러가지 가설을 통해 실험 및 결과를 확인하였다.

내가 한 행동의 결과로 어떤 지점을 달성하고, 어떠한 깨달음을 얻었는가?

- 가설을 통해 실험을 진행하면서 결과에 대한 근거를 명확하게 파악할 수 있었으며, 개선 여지가 있는 부분은 적극적으로 반영하여 리더보드 1위를 하였다.

전과 비교해서, 내가 새롭게 시도한 변화는 무엇이고, 어떤 효과가 있었는가?

- Git 을 적극적으로 활용하여 팀원들의 의견을 파악하여 코드를 수정하였다.
- 팀 협업의 중요성을 깨달았다.

마주한 한계는 무엇이며, 아쉬웠던 점은 무엇인가?

- 시간, 몸 관리를 제대로 하지 못하여 논문 구현을 완벽하게 하지 못하여 성능 향상을 기대할 수 없었다.

한계/교훈을 바탕으로 스스로 새롭게 시도해볼 것은 무엇일까?

- 효율적인 시간 관리 전략을 통해 프로젝트에 적용할 수 있는 논문을 구현할 것이다.

염성현

이번 프로젝트에서 나의 목표는 무엇이었는가?

- 끝까지 최선을 다하는 것과 좋은 성적을 얻는 것이었다.

나는 내 학습 목표를 달성하기 위해 무엇을 어떻게 했는가?

- 대회 기간 동안의 계획을 세웠고 그 계획을 이루기 위해 매일 노력하였다.

나는 어떤 방식으로 모델을 개선하였는가?

- 데이터를 증강하였고 증강 데이터를 정제하였다. 전처리를 위해서 1만 여개의 데이터를 직접 관찰하였고 정제 가능한 표현들을 최대한 정제하였다. 정제의 목표는 주어진 학습 데이터와 유사한 데이터를 만드는 것이었다. 문맥과 상관 없는 html이나 json 관련 표현과 각주나 링크 같은 양식적인 표현들을 정제하였다. 추가로 flag를 만들어 다른 팀원들이 데이터 증강을 용이하게 사용할 수 있도록 하였다.

내가 한 행동의 결과로 어떤 지점을 달성하고, 어떠한 깨달음을 얻었는가?

- 팀이 리더 보드 1위를 하는 데 기여하였다. 가장 크게 깨달은 것은 팀의 위력이었다. 새로고침 기간에 많이 쉬지 않았더니 그 여파로 대회 마지막 기간에 체력적인 한계가 왔었다. 내가 지쳐있을 때 프로젝트 마무리를 위해 끝까지 열심히 해준 팀원들을 보고서 깊은 감명을 받았다.

전과 비교해서, 내가 새롭게 시도한 변화는 무엇이고, 어떤 효과가 있었는가?

- 프로젝트를 적극적으로 리드하였다. 프로젝트 매니저를 직접 맡아서 작업을 수행하였다. 책임을 지는 것을 두려워하지 않았고 팀원들에게 필요한 작업을 적극적으로 부탁하였다. 그 결과 이전 프로젝트에 비해 프로젝트의 통합성이 좋아졌고 프로젝트의 속도감 또한 상승하였다.

마주한 한계는 무엇이며, 아쉬웠던 점은 무엇인가?

- 한계는 Dense Retriever 구현에 있었다. 데이터 증강에 들어가는 모델 학습 시간이 길다보니 Dense Retriever 구현에서 발생한 문제점들을 파고들어서 분석하지 못하였다. 성능이 잘 나오지 않은 것에 대한 원인 분석과 성능 향상을 위한 방법 고안을 끝내 하지 못해서 아쉬웠다.

한계/교훈을 바탕으로 스스로 새롭게 시도해볼 것은 무엇일까?

- 우선, 위의 한계는 시간적인 한계라 아쉬움이 남긴 해도 어쩔 수 없었던 것 같다. 추후에 Dense Retriever를 사용할 기회가 생긴다면 그때 좀 더 연구해보고 좀 더 시도해보겠다. 그리고 앞으로는 프로젝트 리드의 경험을 교훈 삼아 어떤 프로젝트를 하더라도 책임을 마다하지 않고 더 적극적인 자세로 프로젝트에 임할 생각이다.

최동민

이번 프로젝트에서 나의 목표는 무엇이었는가?

- 베이스라인 코드를 직접 커스텀할 수 있는 부분은 할 수 있을 만큼 코드를 제대로 이해하는 것이 목표였다.
- 데이터 측면에서 EDA를 포함해 다양한 분석과 시도를 하는 것이 목표였다.

나는 내 학습 목표를 달성하기 위해 무엇을 어떻게 했는가?

- 베이스라인 코드의 기반이 되는 huggingface document를 많이 참고하였고, 코드의 흐름을 따라가기 위해 많은 시간을 투자하였다.

나는 어떤 방식으로 모델을 개선하였는가?

- 데이터 EDA를 진행하였는데, 구체적으로 데이터 구조를 뜯어보고 ODQA에 적합한 데이터인지 등을 파악하였다.

내가 한 행동의 결과로 어떤 지점을 달성하고, 어떠한 깨달음을 얻었는가?

- 데이터 EDA를 직접 해보고 우려사항을 확인하고 인사이트를 도출하는 경험을 하였다.
- 또한, 데이터가 성능에 직접적인 영향을 끼치는 것을 보면서 데이터를 분석하고 처리하는 과정이 전체 프로세스에서 중요한 과정이란 것을 깨달았다.

전과 비교해서, 내가 새롭게 시도한 변화는 무엇이고, 어떤 효과가 있었는가?

- 전에는 model 구조에 집중하여 바꿔보는 시도를 많이 했다면, 이번에는 데이터를 다뤄보는 경험을 하였고, 이는 전체 프로세스에서 데이터 부분의 중요성을 깨닫게 해주는 효과가 있었다.

마주한 한계는 무엇이며, 아쉬웠던 점은 무엇인가?

- EDA를 통해 마주한 질문들과 우려사항을 다 분석해보지 못한 한계가 있었고, 해결하지 못해 아쉬웠다.

한계/교훈을 바탕으로 스스로 새롭게 시도해볼 것은 무엇일까?

- 데이터를 살펴보고 마주한 질문들과 우려사항들을 잘 기록하고 원인 분석 및 해결을 시도할 것이다.
- 프로젝트에 주도적으로 임하고, 프로세스에 상관 없이 다양한 질문들을 던지고 해결하려고 노력할 것이다.

홍인희

이번 프로젝트에서 나의 목표는 무엇이었는가?

- github을 통해 단순히 code만 업로드하는 것이 아닌, 팀원들과 함께 git flow를 지켜 PR하고 code review 등을 하며 github을 완벽하게 사용하는 것이 목표였다.

- 리더보드 상위권으로 프로젝트를 끝마칠 수 있게끔 최선을 다하는 것이 목표였다.

나는 내 학습 목표를 달성하기 위해 무엇을 어떻게 했는가?

- git flow부터 issue, PR, code review하는 방법 등을 정리하고, 우리 팀 내부 세미나를 열어 발표하며 팀원들에게 github 사용의 중요성을 강조하였다.
- 가설을 세우고 실제 실험을 통해 검증하며 성능을 최대한으로 끌어올릴 수 있게 노력하였다.
- 리더보드로 평가하는 대회 특성상, 어떤 방법들이 가장 성능을 끌어올릴 수 있는지를 생각하고 그에 알맞는 실험을 하였다.

나는 어떤 방식으로 모델을 개선하였는가?

- 강의를 듣고, 여러 가지 추가 자료를 조사하며 가설을 세웠다. 세운 가설과 조사한 자료를 토대로 retriever 실험(TF-IDF와 BM25 / sparse와 dense)을 하며 최종적으로 95%의 정확도로 질문에 알맞는 문서를 가져오는 retriever를 만들 수 있었다.
- 조사한 알고리즘 github를 참고하여 현재 나에게 필요한 부분에 맞게 코드를 작성하였다.

내가 한 행동의 결과로 어떤 지점을 달성하고, 어떠한 깨달음을 얻었는가?

- github에서 PR을 올리면 모두가 code review하는 과정을 통해 모두가 같은 팀 내에서 코드를 완벽하게 이해할 수 있었다.
- 우리 팀이 리더보드 1등을 하는 결과를 얻을 수 있었다. 여러 가지 논문을 읽고 자료 조사를 하며 꼼꼼하게 세웠던 가설들이 실제 결과와 많이 일치하는 모습을 보여주었다. 이를 통해 단순히 많은 실험을 하는 것도 중요하지만, 어떤 결과가 나올지 스스로 예측하고 필요한 실험을 골라서 하는 것 또한 성능을 높이는데 중요한 요소임을 알 수 있었다.

전과 비교해서, 내가 새롭게 시도한 변화는 무엇이고, 어떤 효과가 있었는가?

- 전에는 데이터가 가장 중요하다고 생각해서 데이터 쪽에 집중해서 프로젝트를 진행했었지만, 이번 프로젝트에서는 다양한 논문을 읽으며 모델을 재구성해보고 싶었다. 실제로 프로세스를 정확히 이해하고 모델을 알맞게 재구성하여 데이터 못지 않게 성능을 많이 올릴 수 있었다.

마주한 한계는 무엇이며, 아쉬웠던 점은 무엇인가?

- 욕심일 수도 있지만, 데이터 팀과 모델 팀의 속도가 맞지 않아 모델 팀을 맡았던 내가 데이터 쪽도 살펴보며 마지막에는 이도저도 아닌 과정이 되었다. 앞으로는 다른 파트에 눈을 돌리기보다는내가 맡은 부분만을 최선을 다해 발전시켜야 할 것이다.

한계/교훈을 바탕으로 스스로 새롭게 시도해볼 것은 무엇일까?

- 데이터, 모델 둘 다 놓치지 않고 프로젝트에서 좋은 성능을 낼 수 있도록 노력할 것이다.
- 더 나아가 협업에 필요한 다양한 툴을 더욱 활용할 것이고, 모델 서빙까지 시도할 것이다.