

MRC: Open-Domain Question Answering

Wrap-up Report

NLP-09 조(역삼동불나방)

1

프로젝트 개요

1. 프로젝트 주제 및 목적

- Question Answering(QA)은 다양한 종류의 질문에 대답하는 인공지능을 만드는 연구 분야이다. 그중 Open-Domain Question Answering(ODQA)은 주어지는 지문이 따로 존재하지 않고 사전에 구축되어 있는 knowledge resource에서 질문에 대답할 수 있는 문서를 찾는 과정이 추가되어 질의에 대한 답변을 제시한다.
- 본 프로젝트는 질문에 관련된 문서를 찾아주는 “retriever” 단계와 관련된 문서를 읽고 적절한 답변을 찾거나 만들어주는 “reader” 단계를 각각 구성하고 그것들을 적절히 통합하여 질문에 대한 답변을 해주는 ODQA 시스템을 만드는 것을 목적으로 한다.

2. 프로젝트 환경

컴퓨팅 환경	5인 1팀, 인당 V100 서버를 VSCode와 SSH로 연결하여 사용
협업 환경	Notion, GitHub, WandB, HuggingFaceHub, Google Drive
의사 소통	Slack, Zoom, 카카오톡

3. 프로젝트 구조

a. 데이터 구성

분류(디렉토리 명)	세부 분류	샘플 수	용도	공개여부
train_dataset	train	3952	학습용	모든 정보 공개 (id, question, context, answers, document_id, title)
	validation	240		
test_dataset	validation	240 (Public)	제출용	id, question 만 공개
		360 (Private)		

b. 데이터셋 구조

Feature	설명
id	질문 고유의 id
question	질문
answers	답변에 대한 정보. 하나의 질문에 하나의 답변만 존재함
context	답변이 포함된 문서
title	문서의 제목
document_id	문서의 고유 id

c. 평가 지표:

1) **Exact Match(EM)** : 모든 질문에 대하여 모델의 예측과 실제 답이 정확하게 일치할 때 1점 부여

2) **F1 Score** : 겹치는 단어도 고려해 부분 점수 부여 (참고용)

d. 대회 중 리더보드 평가 기준: Public Score (Test 중 240개의 질문 셋 일부)

e. 최종 평가 기준: Private Score (Test 600개의 질문 셋 전체)

f. 하루 제출 횟수: 10회

2

프로젝트 팀 구성 및 역할

1. 역할

- 전현욱 : 팀 리더, Soft Voting Ensemble, Reader 학습 개선, Retriever 개선
- 곽수연 : Context Preprocessing, Reader 학습
- 김가영 : Reader Negative Sampling 적용, Reader 학습
- 김신우 : DPR Negative Sampling 구현, Reader 학습
- 안윤주 : BM25 구현, DPR Bi-Encoder 학습

3

프로젝트 수행 절차 및 방법

1. 프로젝트 기간

- 2024-02-07 10:00 ~ 2024-02-22 19:00

일	월	화	수	목	금	토
4	5	6	7	8	9	10
	강의 수강			EDA 및 베이스라인 분석		
11	12	13	14	+	15	16
	가설 수립 및 검증					
		데이터 증강				
18	19	20	21	22	23	24
	모델 평가		모델 앙상블 및 최종제출			

2. 활용된 기술 및 라이브러리

- 개발 언어: Python
- 데이터 전처리 및 증강: Pandas, Numpy
- 모델링: PyTorch, HuggingFace
- 성능 분석: WandB

3. 프로젝트 세부 수행 절차

- 1) 2024-02-09 (금)까지 대회 기초 강의 전부 수강 완료
- 2) 서버 수령 후 GitHub 및 작업 환경 설정
- 3) 대회 개요 및 데이터 성질, 베이스라인 코드 분석
- 4) EDA 진행 후, 데이터 전처리 및 분석 내용을 바탕으로 가설 설정
- 5) 모델 선정 및 역할 분담
- 6) 가설 실험 및 검증
- 7) 검증 결과를 바탕으로 최종 방법 선택 후 모델 학습 및 평가
- 8) 평가한 모델 성능 비교 후 Ensemble 진행
- 9) 최종 결과물 제출

4

프로젝트 수행 결과

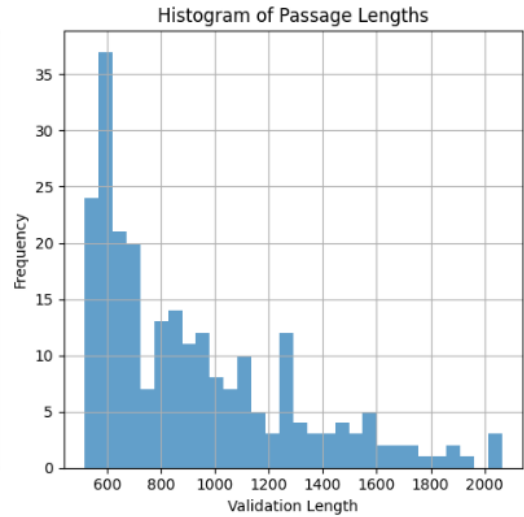
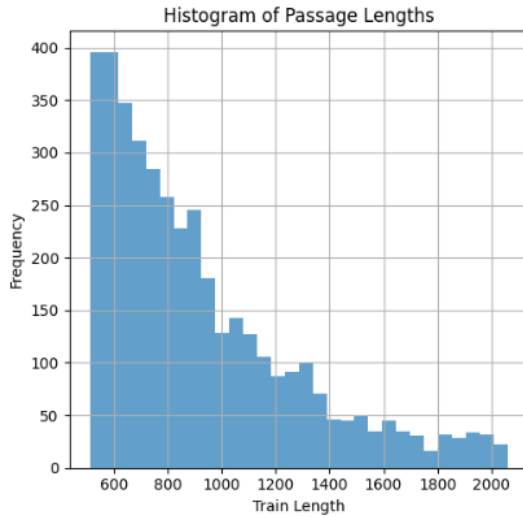
1. 작업 환경 설정

- GitHub : branch 별로 버전 관리
- V100 서버 : 코드 작성 및 모델 학습 (GPU CUDA 버전 : 11.4)
- HuggingFaceHub : Reader 및 Retriever 모델 공유
- WandB : 팀 프로젝트 공간에서 모델 성능 분석

2. EDA

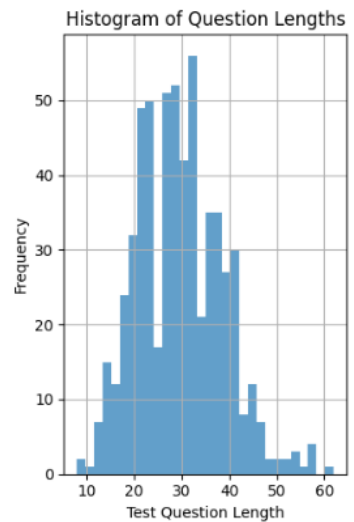
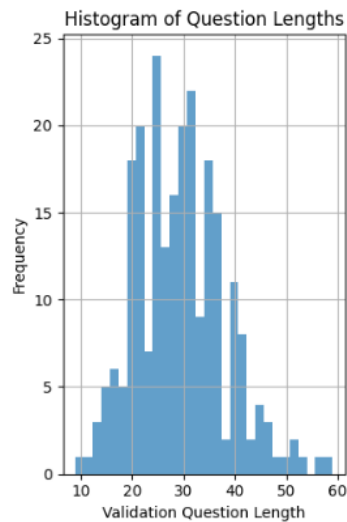
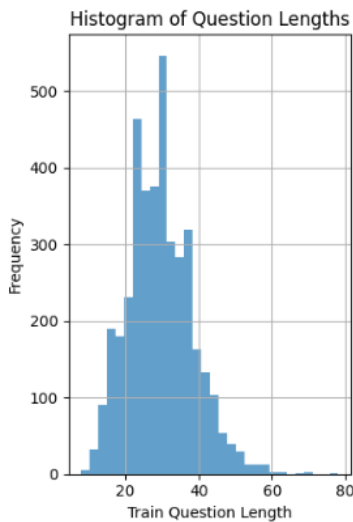
- Train set, Validation set의 Context 확인

Datasets의 Train set과 Validation set은 Context와 Question의 Pair로 구성되어 있으며 Train과 Validation의 Context 길이와 그 분포는 유사함.



- Train, Validation, Test set의 Question 확인

Train과 Validation, Test set의 Question 길이의 분포 역시 유사함.



- Context와 Question 안에 존재하는 특수문자
 - 한문, 일어, 러시아어, 타밀어, 베트남어, 프랑스어 등

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
狂	私	首	果	菩	梯	鉈	弱	球	。	橋	憐	洸	潭	幾	障

- 개행문자: \n \r\n, \n\r\n, \r\n\r\n
- 따옴표: " , '
- 괄호: (),
- 기타 문자: ☞, •, ⑤, □, %, Δ, !, >, ~, ↓, <<, *, **

3. 가설 수립 및 검증

- LG CNS의 KorQuad 데이터셋을 증강 데이터로 사용하면 성능이 오를 것임.
- Reader 학습 시, Validation Score가 EM 기준으로 가장 높은 checkpoint를 저장하도록 설정하는 것이 좋을 것임.
- Reader 학습 시, 데이터에 Negative Sample을 섞으면 성능이 향상될 것임.
- Retrieval에 사용되는 문서 집합의 context를 전처리하면 성능이 향상될 것임.
- Retrieval Embedding을 TF-IDF가 아닌 BM-25와 DPR의 조합으로 사용하면 향상될 것임.
- DPR 학습 시, In-Batch 학습을 사용하면 Retrieval Encoder 모델의 성능이 향상될 것임.
- 특정 토큰라이저를 활용할 때, BM25의 성능을 더욱 향상시킬 것임.

4. 외부 데이터셋 활용

- KorQuad 1.0¹

LG CNS의 AI 연구팀에서 만든 한국어 MRC 데이터셋. 한국어 위키백과(Wikipedia)를 대상으로 구축된 Extractive MRC 데이터이며, 해당 데이터셋은 1,560개의 Wikipedia 문서에 대해 10,645건의 문단과 66,181개의 질의응답 쌍으로 이루어져 있음. 해당 데이터는 학습 데이터 60,407개, 검증 데이터 5,774개, 평가 데이터 3,898개로 구분되어짐.

KorQuad 데이터를 두 가지 방법으로 Reader 모델 학습에 활용하였음. 활용 방법은 아래와 같음.

- 1) 제공 받은 데이터와 외부 데이터 KorQuad를 모두 활용하여 한 번에 모델을 학습
- 2) 데이터를 나눠 기학습된 모델을 파인튜닝 하는 방법. 외부 데이터인 KorQuad로 먼저 학습시킨 모델을 활용하여 제공받은 데이터로 추가적으로 학습

KorQuad 데이터를 아래와 같은 방법으로 Retriever 모델 학습에 활용하였음.

- 1) Dense Passage Retrieval 과정에서 필요한 Bi-Encoder 학습에 KorQuad 데이터 활용. Question과 Context 간 관계를 파악하기 제공받은 데이터의 양이 부족하다고 판단했기 때문. 학습에 소요되는 시간에 제한이 있어 전체의 10%만 사용.

5. 모델 선정

- Baseline 기준으로 성능이 높은 모델 위주로 선정

¹ <https://korquad.github.io/KorQuad%201.0/>

- klue/roberta-large²
- nlpotato/roberta_large-ssm_wiki_e2-origin_added_korquad_e5³
- uomnf97/klue-roberta-finetuned-korquad-v2⁴

6. 모델 학습 및 성능

1) Reader

a) Best Model at Validation Score(EM) Checkpoint

- 리더보드 평가 기준이 되는 Score가 EM(Exact Match)이었기 때문에 학습 시에 EM을 기준으로 Best Model을 저장하도록 설정
- Public Score가 향상되었음 (35.42 -> 38.33)

b) Negative Sampling

- 현재 가지고 있는 Data는 모두 Context 내에 정답이 있다고 가정하는 Golden Passage로만 이루어져 있음
- 정답이 없다는 경우도 만들기 위해 같은 데이터셋 내의 Context를 활용하여 서로 다른 쌍을 만들어 활용
- Positive Sample과 Negative Sample의 비율은 1대1과 1대2로 실험
- 성능 향상이 미미했음

2) Passage Retrieval

a) Preprocessing Wikipedia_documents

- Document 내의 context에 'WWn' 와 같은 특수 기호가 존재함을 확인
- 해당 특수 기호를 제거하고 TF-IDF를 기준으로 Retrieval 했을 때의 성능을 확인
- Public Score가 향상되었음 (45.00 -> 45.83)

b) Topk

- Retrieval 후 context를 몇 개 가져오는지 설정하는 Hyperparameter
- k의 값이 높을수록 가져온 Context 내에 정답이 있는 Context가 포함되는 확률은 높아졌지만, reader의 추론 과정에서 방해가 될 수 있기 때문에 적절한 값이 필요
- k = 25였을 때 가장 성능이 좋았음

c) BM25 (Best Match 25)

- TF-IDF에서 context의 길이까지 고려하는 Sparse Retrieval 적용
- 'rank_bm25' 모듈을 활용하여 구현
- Tokenizer Function으로 여러 방법을 실험

² <https://huggingface.co/klue/roberta-large>

³ https://huggingface.co/nlpotato/roberta_large-ssm_wiki_e2-origin_added_korquad_e5

⁴ <https://huggingface.co/uomnf97/klue-roberta-finetuned-korquad-v2>

- 띄어쓰기 (split)
- MeCab
- klue/bert-base
- monologg/koelectra-base-v3-finetuned-korquad
- monologg/koelectra-base-v3-finetuned-korquad의 성능이 가장 높았음
- Public Score가 대폭 향상되었음 (45.00 -> 60.83)

d) DPR (Dense Passage Retrieval)

- 3가지 방법론을 고려
 - Bi-Encoder
 - Pretrained Bert와 Pretrained Roberta 구조의 Encoder를 사용해서 비교
 - Bert의 성능이 좋았음
 - Single Encoder
 - [SEP] 토큰으로 Question과 Context를 구분해서 비교
 - Inference 속도가 매우 느리기 때문에 채택하지 않았음
 - ColBert
 - 계산 속도를 대폭 향상시킨 Bi-Encoder
 - 시간 부족으로 적용을 못 했음
- 최종적으로 Bi-Encoder 구조를 사용
- Bi-Encoder 학습 시 두 가지 방법으로 학습
 - Negative Sampling
 - Golden Passage와 함께 Negative Sample을 추가하는 방법
 - Negative Sample은 Random Sampler와 Sentence Transformer를 바탕으로 유사도가 가장 낮은 Sample을 1대1 비율로 선택
 - In-batch Negative
 - Batch 내의 Golden Passage 제외한 모든 Sample을 Negative Sample로 간주하고 학습하는 방법론
- In-batch negative 방법이 가장 좋았음
- TF-IDF 보다 성능이 약간 향상되었음

e) Sentence Transformer Retrieval

- Sentence Transformer로 Question과 Context 쌍을 학습해서 Retrieval 시도
- Negative Sample도 Random하게 1대1 비율로 선택
- 성능 향상이 미미했음

f) BM25 + DPR

- BM25와 DPR을 통해 나온 Document Score의 가중 평균값을 통해 Retrieval
- 논문의 실험 결과를 바탕으로 채택한 방법론으로, 정확한 단어 일치와 의미적

유사도 모두를 반영하도록 하는 목적에서 실험

- 가중치 비율을 여러 번 실험 결과 5대5 비율이 가장 좋았음
- 성능이 가장 좋았던 Retrieval 방법이었음

8. Ensemble 실험


- **Ensemble 모델 선정 기준** - Public score가 높으며 예측 경향성이 다른 모델 위주
- **Soft Voting Ensemble** - n_best_prediction 파일에서 포함된 정답의 Probability를 기준으로 Soft Voting 적용

9. 최종 결과

[Public Score] Exact Match: 66.6700 / F1 Score: 77.1700

4 (-)	NLP_09조		66.6700	77.1700	40	1d
----------	---------	---	---------	---------	----	----

[Private Score] Exact Match: 63.6700 / F1 Score: 74.2200

5 (1 ▾)	NLP_09조		63.6700	74.2200	40	2d
------------	---------	---	---------	---------	----	----

[최종 채택한 방법론 및 모델]

- 데이터 전처리 및 증강
 - 기존 데이터 + KorQuad v1 dataset
- 모델
 - 최종 Reader 모델
 - 1) klue/roberta-large only with Golden Passage
 - 2) klue/roberta-large only with Negative Samples
 - 최종 Retrieval 모델
 - 1) BM25 + Bi-Encoder DPR (5:5)

1. 팀 회고

- **잘했던 점**
 - Baseline 코드를 Jupyter 노트북 파일로 나눠 학습하면서 코드의 흐름을 잘 따라가며 실험했다.
 - 강의를 바탕으로 여러 논문을 찾아보면서 성능을 향상시킬 수 있는 방법론을 채택하고 실험해 보았다.
 - 주어진 데이터 및 외부 데이터셋을 어떤 방법으로 활용할지 여러 Case를 나눠서 실험해 보았다.
- **아쉬웠던 점**
 - Baseline에 대한 이해의 속도가 느려서 이전 프로젝트보다 진전이 더뎠다.
 - 하이퍼파라미터 튜닝을 Grid Search를 통해 최적의 값을 도출하려고 시간을 들여서 실험해 보았는데, default 값이 제일 좋았어서 시간 대비 비효율적이라는 것이 느껴졌다.
 - 우리가 선택한 모델이 Private이 공개되고 최적의 모델이 아니라는 것을 알았는데, 여전히 일반화가 잘 된 모델에 대한 이해도가 낮은 것 같아서 아쉬웠다.
 - Faiss를 제대로 활용해 보지 못해서 아쉬웠다.
- **개선 사항**
 - 하이퍼파라미터 튜닝에 sweep을 사용하는 방법이 있었는데 이번 프로젝트에는 사용하지 못해서 다음 프로젝트에서는 sweep을 활용해서 최적의 하이퍼 파라미터 발견을 시도해 보고 싶다.
 - Git Flow 방법을 채택하여 프로젝트를 진행하고 있다. 그래서 하나의 실험이 끝나면 Pull Request를 하게 되는데 이번 프로젝트에서는 따로 코드 리뷰 없이 merge 했다. 다음 프로젝트에서는 코드 리뷰를 꼼꼼하게 해보고자 한다.
- **프로젝트 통해 배운 점 및 시사점**
 - 2-stage 프로젝트이다 보니, 베이스라인 코드가 여러 코드로 분산이 되어 있었는데, 대회를 진행하면서 패키지 모듈화에 대한 이해가 깊어진 것 같다.
 - Embedding 방법과 방법에 따른 장단점을 프로젝트를 통해 직관적으로 학습할 수 있었다. 개념적으로만 알던 방법론을 실질적으로 적용하면 어떤 결과가 일어나는지 체득할 수 있는 기회가 되었던 것 같다.