P3 – MRC

(Machine Reading Comprehension)

Wrap up Report

2021/10/12 ~ 2021/11/04

T2020 김다인

boostcamp

목차

I. 프로젝트 목표

이번 프로젝트 또한 여느 때와 마찬가지로 개인적인 측면에서는 다양한 실험과 구현을 통한 성장에 목표를 두었고, 팀의 측면에서는 탐구나 실험의 결과를 원할히 공유하여 모든 팀원들이 현재 팀의 상황을 훤히 알 수 있도록 하는 것에 목표를 두었다. 또한 하나의 코드로 진행되도록 브래치를 관리하고 pull request를 적극적으로 활용하기로 정하였다.

II. 목표 달성과 상호 발전을 위한 나의 노력

1. BM25 구현

- 제공 받은 베이스라인 코드에선 TF-IDF를 이용해 retrieval의 score 계산을 하고 있었다. TF-IDF의 개념을 바탕으로 발전시킨 BM25를 직접 구현하여 retrieval를 개선하고자 하였다.
- 이미 만들어진 BM25패키지를 사용하지 않고 사이킷런 패키지의 TfidfVectorizer 만을 이용해 BM25를 구현하였다. 이 과정은 두 계산 방식의 차이점과 관련된 코드를 정확히 이해해야 가능했기에 적용하며 BM25와 TF-IDF의 구조에 대해 보다 깊이 알 수 있었다. 때문에 성장의 측면에서 Elastic Search의 BM25를 바로 이용하지 않은 점이 전혀 후회되지 않았다. 이는 대회의 우선적인 목표가 점수가 아니라 공부와 성장인 점에 부합한다고 생각한다.
- 기존의 TF-IDF는 정답이 들어있는 문서를 가져오는 성능이 topk 가 1일 때 0.25, 10일 때 0.63이었지만, BM25는 성능이 topk 1에서 0.42, 10에서 0.77의 향상된 성능을 보였다.

2. Elastic Search 세팅 및 개선

- 오피스 아워를 통해 현재도 주로 사용되는 좋은 성능의 문서 검색 엔진인 Elastic Search를 알게 되었다. Scoring에 BM25를 default로 사용하고 있었기에 BM25를 구현하고 몇가지 실험을 해본 이후 바로 Elastic Search 구현을 다음 목표로 설정하였다.
- 세팅 과정에서 root 로 실행이 되지 않는 약간의 어려움이 있었으나 daemon을 이용해 이 문제를 해결할 수 있었다.
- 구현 후 analyzer의 filter를 바꾸는 실험을 통해 topk 1: 0.71, 10: 0.89까지 성능을 올릴 수 있었고, Elastic Search의 위력과 인덱스를 이용한 속도를 체감할 수 있었다.
- 이에 그치지 않고 사람의 검색 방식을 모방해 부가 점수를 주고자 하였다. 보통 우리는 검색할 때, 문장 전체를 검색 엔진에 검색하지 않고 몇 개의 결정적인 키워드를 주로 이용한다. 이를 구현하기 위해 Pororo의 ner tagging을 이용해 태깅이 가능한 고유 명사들을 따로 추출하였다. 그리고 Elastic Search의 bool query 기능을 이용해 should 문안에 이 고유명사들을 넣어줘 이 명사들이 들어간 문서에는 추가 점수를 주도록 구성하였다.
- 그러나 위 방법을 적용하면 ner tagging에 생각보다 시간이 오래 걸려 retrieval의 성능 평가나 inference에서 너무 많은 시간이 걸리게 되었다. 이는 question에 대해 ner tagging을 미리 적용한 csv파일을 미리 만들어 두는 것으로 해결할 수 있었다.
- 이를 통해 최종적으로 retrieval의 성능을 topk 1: 0.73, 10: 0.92까지 올릴 수 있었다.

3. Wiki data 전처리

- 찾아와야 하는 passage들은 모두 wikipedia에서 가져온 것으로, 여러가지 특수 문자들이 섞여 들어가 있었다. 이 중 정답으로 쓰이지 않는 것은 대부분 retrieval와 reader의 성능을 억제한다고 판단했다.
- 토론 게시판에서 다른 캠퍼님이 train, validation data의 정답 구문에 쓰이는 특수 문자를 올려주신 글을 참고하여 '₩n', '#'과 같은 정답에 쓰이지 않는 특수 문자를 제거하였다.

- 이 전처리를 적용해 retrieval에선 2퍼센트 정도, reader에선 5퍼센트 정도의 큰 점수 향상을 경험할 수 있었고, nlp에서 전처리의 중요성을 느낄 수 있었다. 이는 대회가 끝난 후 교수님이 피드백을 주시며 잘한 부분으로 칭찬을 주신 부분이기도 해서 그 중요성을 잊을 수 없을 것 같다.

4. Stratified K-fold validation set 구축 및 코드 공유

- K-fold에 대한 신뢰는 대회를 진행하면 할수록 더욱 커져만 가는 것 같다. 특히 이번 대회의 베이스라인은 validation 데이터가 240개의 매우 적은 수로 주어졌기에 좋은 validation 데이터가 절실했다. 따라서 이번에도 K-fold validation set 제작과 기존의 코드를 K-fold set을 이용해 동작할 수 있도록 바꾸는 작업을 맡아 진행했다.
- 이번에 K-fold set 구성 단계에서 가장 크게 고민한 점은 "정답passage가 같은 질문들을 같은 fold group에 넣을 것인가?" 의 여부였다. 이렇게 구성한다면 validation 단계에선 항상 train때엔 보지 못한 passage를 접하게 되고 그 안에서 정답을 찾아야할 것이었기에 처음엔 이 방법이 좋다고 생각해 적용했다. 그런데 validation score가 기존보다 많이 낮게 나왔고, 곰곰이 고민해보니 결국 inference에서도 이미 본 passage가 나오지 않는 것이 아니었기에 train의 다양성을 줄이는 일이 된다고 결론을 내릴 수 있었다.
- 따라서 최종적으로 정답 passage의 길이를 기준으로stratifty하게 나눠주어 다양한 길이의 passage에서 정답을 가져오고 검증하도록 5-fold set을 구성하였다.
- 저번 대회에서 다짐했던 각 fold를 csv 파일로 만들어 저장하는 과정도 수행하여 모두가 같은 5-fold set에서 실험을 수행하였다. 이렇듯 재현에 각별히 신경을 썼기에 서버가 남는 사람들끼리 같은 실험을 각자 다른 fold 순번으로 돌려 시간을 절약할 수 있었다.

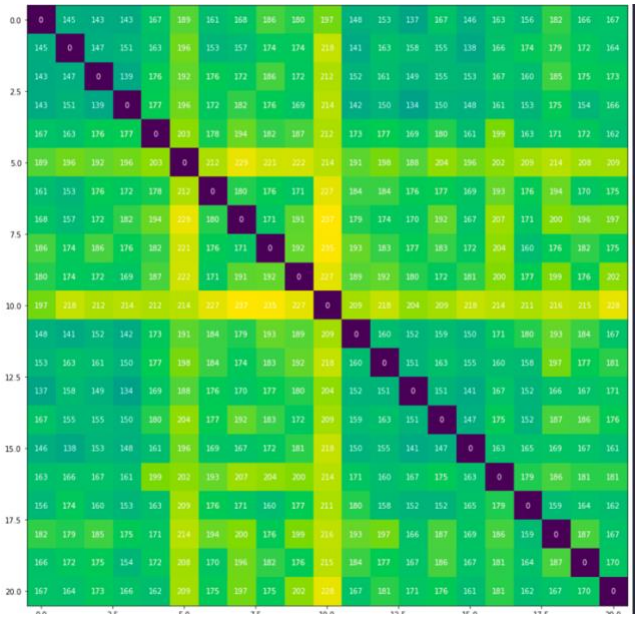
5. Question Masking 실험

- 저번 대회에서도 적용해보았던 masking을 이번에도 적용해보았다. Question에 masking을 적용하면 문제의 난이도가 올라가 더욱 좋은 성능을 기대할 수 있을 것이라고 판단하였기 때문이다.
- Question별로 50퍼센트의 확률로 masking 여부를 결정하고 masking을 하기로 결정된 question은 mecab을 이용해 명사만 추출하여 그 중 하나를 랜덤하게 뽑아 가리는 방식으로 진행하였다,
- 실험 결과 아쉽게도 validation score와 LB score가 1~2점 하락하였다. 일단 각 epoch 마다 dynamic 하게 masking이 적용되게 구현하지 않은 점이 가장 큰 아쉬움이며, 다른 이유로는 정답을 맞추는 데에 큰 역할을 하는 명사를 가리게 되어 어쩔 수 없는 성능 저하가 일어나지 않았을까 추측된다.

6. Ensemble

- Inference의 결과로 제출용 predictions.json 파일과 nbest_predictions.json가 나오게 되는데, predictions.json 에는 예측 text가 하나씩 기록되어 있는 데에 반해 nbest_predictions.json에는 상위 20개의 에측 text와 함께 start_logit, end logit, score가 기록되어 있었다.
- 이 nbest_predictions.json의 score를 이용해 score를 더하는 방법으로 5-fold set 각각의 inference 결과를 앙상블할 수 있었고, 그 결과 1fold 단일과 비교해 public 리더보드 스코어를 5점 정도 올릴 수 있었다.
- 대회 마지막 날 좋았던 결과들을 위와 같은 방법으로 앙상블하려 했으나, 아쉽게도 몇 명의 팀원들이 제출했던 결과물의 nbest_predictions.json을 가지고 있지 않았고, 있더라도 찾는 데에 많은 시간이 걸린다고 판단해 대회 사이트에서 쉽게 다시 다운받을 수 있는 predictions.json만을 Hard voting의 방법으로 사용하기로 하였다.
- Voting의 결과가 동률인 경우에는 Public LB score가 높은 쪽이 승리하도록 하는 등의 후처리를 추가하였다.

- 앙상블 시 모델의 추론 결과가 다양하면 다양할수록 더 큰 성능 향상 효과가 있는 점을 이용해, 앙상블의 재료 선정에서 점수보다 다양성을 더욱 우선에 두었다. 따라서 각 파일별 600개의 추론 결과 중 상이한 것의 개수를 아래와 같이 시각화했다.



- K-fold 실험의 단일 제출물 뿐만 아니라 각 재료 5개씩도 넣어보았는데, 생각보다 상이한 개수가 다른 것과 차이가 없어 K-fold의 단일 제출물은 앙상블에 사용하지 않고 대신 재료 5개씩을 전부 재료로 사용하는 방법을 택했다.
- Public LB score가 낮았던 모델도 더하면 더할수록 점수 개선에 효과가 있었기에 최종 제출물은 20개 이상의 제출 파일을 합친 것으로 결정하였다. 특히 모델의 구조가 달랐던 것은 반드시 넣었는데, 이를 통해 다양성의 중요성을 다시금 느낄 수 있었다. 이 방법을 통해 마지막 날 6점 가량 점수를 올릴 수 있었다.

III. 결과 및 느낀 점

- P2 대회와 동일한 **public 2등, private 2등**이라는 좋은 결과를 얻었다.
- 이번에도 역시 좋은 팀원들 덕분에 이러한 결과를 얻을 수 있었다고 생각한다. 2개의 대회가 연속되기도 했고 이번 대회는 4주라는 긴 기간 동안 진행되어 모두가 지쳐 있었지만, 힘든 만큼 하나로 똘똘 뭉쳐 이겨낼 수 있던 것 같다.
- 저번보다 git을 더욱 알차게 이용했는데, 모두가 pull request를 통해 하나의 코드로 작업을 진행했으며, 모듈화를 생각하고 작업했기에 다른 사람들과 코드를 합치는 과정이 매우 수월했다.
- 확실한 성능 확인이 중요함을 깨닫고 초반에 retrieval의 성능 평가 함수를 보수하고 reader에서 사용할 validation set을 구성하는 데에 집중했는데, 이후의 실험들을 매끄럽게 진행하고 평가할 수 있었던 점이 좋았던 것 같다.

IV. 한계 및 개선해야 할 점

- 베이스라인이 기존에 경험했던 대회들에 비해 이해하는 데에 난이도가 있어 첫 주에 다양한 시도를 못해 본 것이 아쉽다. 물론 첫 주는 이론을 공부하는 데에 많은 시간을 쏟아 어쩔 수 없었지만, 어차피 베이스라인의 어려움은 해결했어야 하는 과제였고, 실제로 2주차부터 조금씩 적응해 나갔기에 이 과정을 조금 더 빨리 시작했으면 어땠을까 하는 생각이 든다.
- Reader 부분에서 보다 다양한 실험을 하지 못한 것이 아쉽다. 난 주로 retrieval 파트를 맡아 개발했고 reader에선 K-fold나 question masking을 맡았는데, 시간만 허락했다면 리더 모델의 구조를 개선하는 등의 다양한 실험을 해보았을 것 같다.