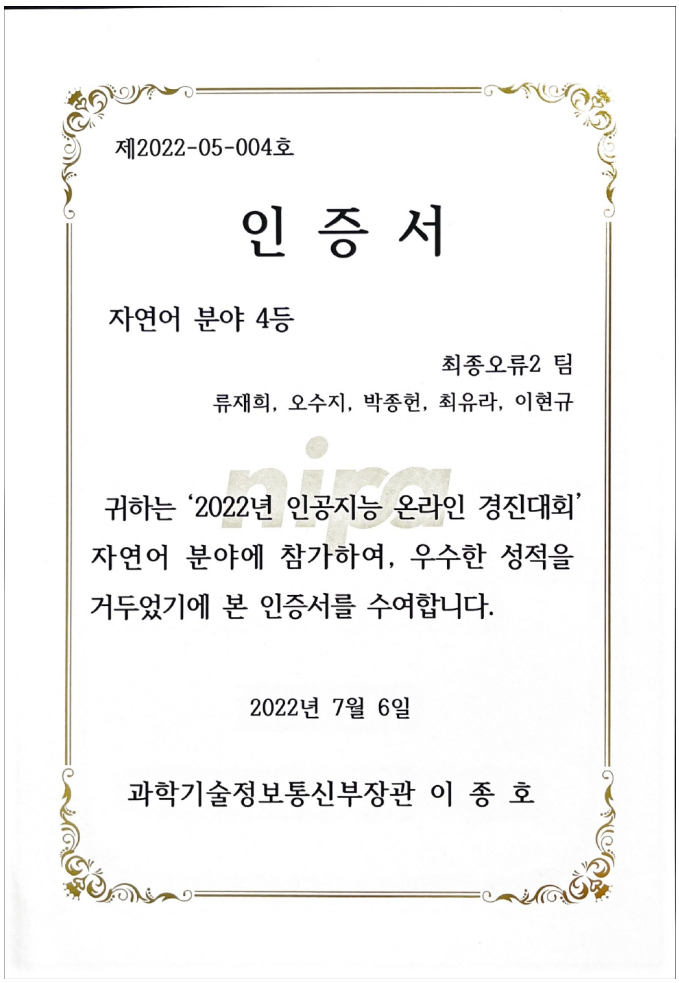
2022 Online AI Competition (Machine Reading Problems for Efficient Document Retrieval)

Result

성과

총 26개의 Team 중 4위를 차지함.



제2022-05-004호

인 증 서

자연어 분야 4등

최종오류2 팀

류재희, 오수지, 박종헌, 최유라, 이현규

귀하는 '2022년 인공지능 온라인 경진대회' 자연어 분야에 참가하여, 우수한 성적을 거두었기에 본 인증서를 수여합니다.

2022년 7월 6일

과학기술정보통신부장관 이 종 호

GitHub Link

https://github.com/LeeHyeonKyu/aichall_2022_mrc/tree/main

Competition Background

대회 개요

과학기술정보통신부 주관 사업화 지원 경진대회임. (선발 시 사업화 지원금 3억 원 등 지급)

총 10개의 과제 중 '문서 검색 효율화를 위한 기계독해 문제'를 선택함. Context내에서 주어진 Question에 대한 정답을 제시하는 형태의 대회임.

Data 형태

전체 Data는 `json` 형태의 File로 구성되어 있음.

모든 문서는 한국어로 구성되어 있음.

각 Quetsion에 대해서는 1개의 Context와 1개의 Answer가 주어지며, Answer가 시작되는 Index에 대한 정보가 포함되어 있음.

Baseline

Extractive한 형태의 문제로 접근하여, 각 Token이 Start 혹은 End Position일 확률을 출력하도록 구성되어 있었음.

내부적으로 Text Base Index를 Token Base Index로 변환하는 Logic이 구성되어 있음.

Max Length에 의해 Gold Text가 잘리는 경우 가장 마지막 Token을 예측하도록 구성되어 있음.

Detail

한 개의 주제에 대해서 N개의 Context와 Question, Answer가 존재함.

질문에 대한 답변을 할 수 없는 경우 Answer는 공백으로 표현되며, `is_impossible`이라는 Flag로도 표현됨.

Solution Idea

Start, End Position Code 재구성

- Why?

  Baseline에서는 Answer Text가 Max Length를 벗어나는 경우 일괄적으로 Max Length를 정답 Position으로 설정함.

  위 과정으로 학습을 할 경우, 무의미한 Token을 정답 Token으로 판단하고 학습하도록 유도됨.

- How?

  Tokenized Context가 Max Length를 넘어서는 경우를 고려할 수 있도록 Overflow여부를 확인하고, 일정한 Doc Stride만큼 Overlap되도록 Tokenizing Code를 구성함.

  Offset을 기준으로 Answer Text의 Position을 추적하고, 이에 대해 정확히 Labeling하도록 구성함.

- So?

  위 라벨링 로직만 변경 했음에도, Public Score 0.50수준(Baseline)에서 0.58 수준으로 성능이 향상됨.

Backbone Model 실험

- Why?

  기존의 Electra Model보다 적합한 Model이 많고, 성능에 큰 변화가 있으리라 기대함.

- How?

  Klue-Roberta, XLM-Roberta, Splinter 등 다양한 Model에 대한 성능 실험을 수행함.

- So?

  별다른 학습 테크닉이 없음에도, 0.65 수준으로 성능이 향상됨.

Question Shuffle Augmentation

- Why?

  Data의 생성 규칙을 확인해본 결과, 특정 주제 아래에 Context가 있고 각 Context마다 질문이 딸려있음을 확인함.

  답변이 불가능한 질문은 약 20% 정도 존재했으며, 80%의 데이터는 답변이 가능한 Data임을 확인함.

  Context 내에서 답을 할 수 없는 질문은, 타 Context에서 답변을 할 수 있거나, 일부 어휘의 수정을 통해 답변을 할 수 없도록 만든 Data임을 확인함.

  즉, 어휘나 문맥에 대한 디테일한 이해가 중요하다고 판단함.

- How?

동일 주제 내에 답변이 가능한 Context와 질문을 섞음으로써, 답변을 할 수 없는 Data를 생성해냄.

- So?

  Augmentation을 통해 Data의 양을 약 10배 가량 증가시킬 수 있었으며, 학습 시간 등을 고려하여 3배로 증강하여 사용함.

  0.11%p 정도 성능 향상을 얻어, 0.76 수준을 달성함.

Add Casual Convolutional Layer

- Why?

  앞선 MRC 과제에서 수행했던 방법이며, 동일한 이유로 실험을 진행함.

- How?

  Backbone Model의 Output을 N개의 Convolutional Layer에 통과시켜, 인근 Token과의 연산을 하도록 구성함.

- So?

  0.06%p 정도의 성능 향상을 얻어, 0.82 수준을 달성함.

Dynamic Augmentation 수행

- Why?

  Question에 대한 Paraphrasing을 통해 Augmentation을 적용해 Data의 다양성을 늘리고자 함.

  Random Masking을 통해 Question 및 Context의 일부 Token을 Masking하여, Data의 다양성 및 난이도를 증가시키고자 함.

  앞선 Question Shuffle Augmentation까지 수행하기 위해 Dynamic한 Augmentation Code를 구성하고자 함.

- How?

  KoGPT2와 Pororo Library를 활용해, Question의 다양한 Paraphrasing Version을 생성함.

Tokenizing 이후, 일부 Token을 `[MASK]` Token으로 변경하도록 구성함.

Dynamic한 Augmentation을 위해서 `Dataset` Class의 `__getitem__` 호출 시 동작하도록 구성함.

- So?

  0.02%p 정도의 성능 향상을 얻어, 0.84 수준을 달성함.

KorQuad SDS Head

- Why?

  KorQuad에서 좋은 성과를 거둔 SDS 팀의 Idea 중, CNN 아키텍처가 도움이 될 것이라 판단함.
  (Ref. https://www.youtube.com/watch?v=ovD_87gHZO4)

  Token의 Length 축으로 연산을 수행하던 Casual Convolutional Layer와는 상반된 연산으로서, 이 연산이 새로운 정보를 추가해 줄 수 있으리라 판단함.

- How?

  SDS Head 라는 명칭으로 Module을 구성하고, 이를 Model 연산 내부에 추가함으로써 모델을 구현함.

- So?

  0.02%p 정도의 성능 향상을 얻어, 0.86 수준을 달성함.

Joint Token Logit

- Why?

  Model의 예측 결과가 터무니 없이 길거나, Start Token보다 End Token이 앞에 나와 공백이 되는 경우가 있음을 확인함.

  Start와 End의 Logit의 각각의 Argmax를 구하는 것은, 두 Token의 연관 관계를 고려하지 못한다고 판단함.
  (Ref. https://tech.kakaoenterprise.com/144)

Start와 End Token의 Joint Logit을 구하고, 이를 학습에 활용하면 이에 대한 관계를 학습할 수 있으리라 판단함.

Data의 분포를 살펴보았을 때, Gold Text의 Length가 비교적 짧으며, 마찬가지로 End와 Start Token 사이의 거리가 대체로 짧음을 확인함.

Train Data를 기반으로 한 Validation Score와 Public Score가 매우 유사한 경향성을 보이므로, Train의 Token Distribution을 활용할 수 있으리라 판단함.

- How?

  Start Token과 End Token의 Logit을 행렬 연산함으로써 2차원 행렬을 얻어냄.

  얻어낸 2차원 행렬을 1차원으로 펼치고, 정답에 해당하는 Start와 End Token의 위치를 기반으로 Loss를 구하도록 설계함.

  Prediction 시에는, Train Data의 Answer Text에 대한 Token Distance Distribution을 활용함.

  End Token과 Start Token의 Topk를 얻고, 위 확률 분포를 활용해 가중치를 주는 방식을 사용함.

- So?

  0.02%p 정도의 성능 향상을 얻어, 0.88 수준을 달성함.