

최종보고서

전기컴퓨터공학부 정보컴퓨터공학전공

신우창(201424479)

이태오(201424513)

김윤호(201424427)

과제명 : 딥러닝과 검색엔진을 활용한 질의응답시스템

지도교수 : 권 혁 철

목차

1. 과제의 목표	3
2. 요구 조건 및 제약사항 분석.....	3
2.1. 요구조건.....	3
2.2. 제약사항	4
2.3. 대책	4
3. 설계 상세화	4
4. 실험환경 및 데이터셋.....	10
5. 실험결과	11
6. 산업체 멘토링 내용 반영 결과.....	12
7. 어려웠던 점 및 결론	12

1. 과제의 목표

딥러닝과 검색엔진을 활용한 질의응답 시스템

- 제한된 도메인에서 정보검색 기술을 바탕으로 사용자의 질문에 응답하는 한국어 질의응답 시스템을 개발한다.
- 검색 엔진으로 사용자의 질문의 주제에 해당하는 위키피디아 및 나무위키 문서를 검색하고, 딥러닝을 이용하여 위키피디아 및 나무위키 문서 내에서 해당하는 답변을 추출하는 것을 목표로 한다.

2. 요구 조건 및 제약사항 분석

2.1. 요구조건

1. User Interface(질의)

클라이언트 프로그램은 사용자가 질문을 하고 답변을 받는 과정에 있어서 여러가지 편의성을 제공하고, 사용자는 클라이언트 프로그램의 UI를 통해 제한 없이 질문을 입력 할 수 있어야 한다.

2. 검색엔진

검색엔진은 사용자의 질의에 따라 한국어 위키피디아의 모든 문서에서 관련 정보가 있는 텍스트를 고속으로 검색하여 출력하여야 한다.

3. 질의응답 모델 구현

질의 응답 모델은 자연어 처리 모델의 자연어 처리 결과를 입력으로 받아 사용자의 질의에 대해 가장 높은 응답을 출력해야 한다.

4. User Interface(응답)

사용자는 클라이언트의 UI를 통해 질의에 대한 응답을 확인 할 수 있어야 한다.

5. 성능개선

Open 도메인 질의응답 시스템의 정확도를 개선시킨다.

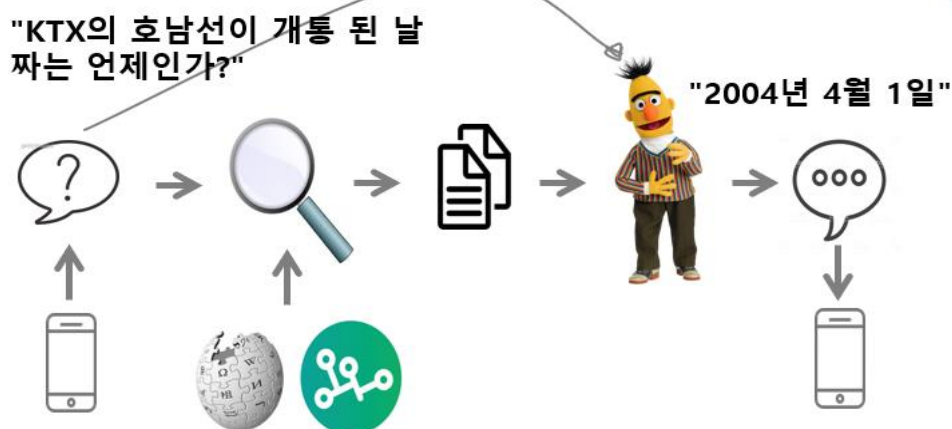


그림 1)

2.2. 제약사항

1. 한국어 질의 응답 시스템

과제의 목표에 따라 질의 응답 시스템의 언어처리는 한국어로 제한된다.

2. 검색엔진

연구실에서 제공하는 미리내 검색엔진을 개량하여 사용한다.

3. 성능

기존의 Bert 모델보다 더 높은 정확도를 요구한다.

2.3. 대책

1. 제약사항 1. 에 대한 대책

한국어 Machine Reading Comprehension을 위해 만든 dataset Korquad를 사용한다.

2. 제약사항 2. 에 대한 대책

대용량 검색 처리를 위한 inverted index 알고리즘과 검색 키워드에 가장 부합하는 문서를 검색 결과 최상위에 배치하는 알고리즘 TF-IDF을 사용하여 구현한다.

3. 제약사항 3. 에 대한 대책

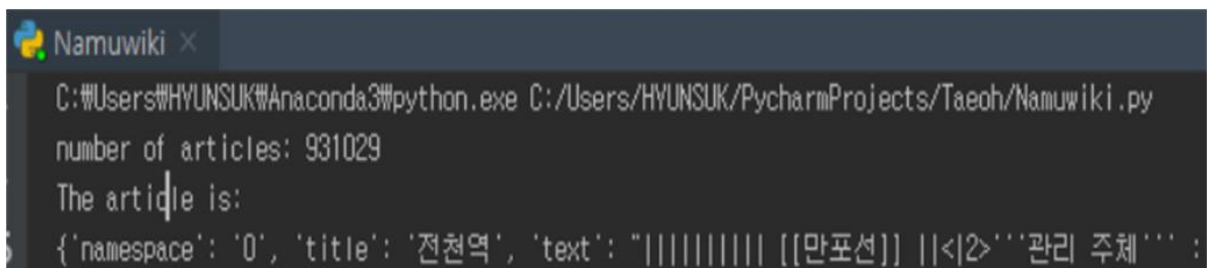
Start와 end의 두 개의 Prediction Layer를 구성하여 확률을 구한 후, 가장 큰 확률을 정답으로 추출한다.

3. 설계 상세화

a. 데이터 전처리

1) Read Dump file

Reading DB Dump File Python의 json 라이브러리를 이용해 NamuWiki의 dump파일을 읽어온다.



```
Namuwiki x
C:\Users\HYUNSUK\Anaconda3\python.exe C:/Users/HYUNSUK/PycharmProjects/Taeoh/Namuwiki.py
number of articles: 931029
The article is:
{'namespace': '0', 'title': '전천역', 'text': "||||||| [[만포선]] ||<|2>''관리 주체'' :
```

2) Data Exploration

NamuWiki의 dump파일의 크기와 각 문서의 최소 길이와 최대길이를 구한다.

```
min text size: 0
max text size: 426441
!
#redirect 느낌표
```

3) Test Parsing

Dump파일이 잘적재되었는지 몇몇 기사의 title, text를 출력한다.

```
#redirect 느림표

!!아앗!!
[[파일:3444050440.jpg]]
([[신 세계수의 미궁 2]])에서 뜬 !!아앗!!
{{(*! !!あっと!!)}}

[[세계수의 미궁 시리즈]]에 간접으로 등장하는 대사, [[세계수의 미궁 2 제왕의 성배|2편 제왕의 성배]]부터 등장했으며, 훌륭한 [[사양 플러그]]의 예시이다.

세계수의 모험가들이 탐험하는 던전인 수해의 구역구역에는 채취/발해/채굴 포인트가 있으며, 이를 위한 채집 스킬에 투자하면 제한된 채집 기회에 보다 큰 큰

1. 채집을 캐릭터들로 이루어진 약한 파티(ex: [[레인저(세계수의 미궁 2)|레인저]]) 5명)가 수해에 입장한다.
1. 월드 전투를 회피하면서 채집 포인트에 도착해 열심히 아이템을 캐는 중에...
1. '''!!아앗!!''' --라클레시아가 나타났다!--
'''이때 등장하는 것은 [[FOE(세계수의 미궁 시리즈)|FOE]]는 아닌 일단 월드 플레이어지만, 훨씬 위 층에 등장하는 강력한 플레이어인 선 텀을 빼앗겼다!'''
1. '''[[오일 죽음|죽일]]'''(hage)

작품마다 !!아앗!!의 세세한 모습은 다르다. 그 악랄함은 첫 등장한 작품이자 시리즈 중에서도 불친절하기로 정평이 난 2편이 절정이었었는데, 그야말로 위의 [[세계수의 미궁 3 성배의 내방자|3편]], [[세계수의 미궁 4 견승의 거산|4편]]에는 숨통이 트이게도 채집 중 낮은 확률로 "좋은 아이템을 얻을 수 있을 것 같"
[[신 세계수의 미궁 밀레니엄의 소녀|신 세계수의]] [[신 세계수의 미궁 2 파프니르기사|미궁 시리즈]], 그 이후에 나온 최신작 [[세계수의 미궁 5 오현 신화]]

세계수 시스템을 기반으로 한 [[패러소나 시리즈]]와의 클라보 작품 [[패러소나 0 세도우 오브 더 러버런스]]에도 물론 등장. 3, 4편과 같이 파워 스킷에서 채

여담으로, 2편에서 채집 도중 !!아앗!!이 볼 확률은 [[http://www.atluspnet.jp/topic/detail/2101]]고작 1%였다고 한다.) 낮아보이는 확률이어도 플레이 중 한

그 기원은 1인칭 던전 크롤러의 원조 [[위저드리]]에서 함정을 건드렸을 때 나오는 대사 Oops! (おっと!)라고 한다.

[[분류:세계수의 미궁 시리즈]]
```

4) Preprocessing with RegEx

Regular Expression을 이용하여 불필요한 문자, 기호들을 제거해준다.

11!아앗!!

신 세계수의 미궁 2에서 뜬 아앗

세계수의 미궁 시리즈에 전통으로 등장하는 대사. 2편 제왕의 성배부터 등장했으며, 훌륭한 사양 플러그의 예시이다.

세계수의 모험가들이 탐험하는 던전인 수해의 구역구역에는 채취불채취용 포인터가 있으며, 이를 위한 채집 스킬에 투자하면 제한된 채집 기회에 보다 큰 이득을 챙길 수 있다. 그러나 분배할 수 있는 스킬은 한정되어 있다.

1. 채집용 캐릭터들로 이루어진 약한 피ilex 레인지 5명가 수해에 입장한다.
1. 월드 전투를 회피하면서 채집 포인트에 도착해 열심히 아이템을 캐는 중에...

1. 아앗
이때 등장하는 것은 F0E는 아닌 일단 월드 플러그지만, 훨씬 위 층에 등장하는 강력한 플러그이며 선 턴을 뺏긴다

1. 띄워hege

작품마다 아앗의 세세한 모습은 다르다. 그 약탈하는 첫 등장한 작품이자 시리즈 중에서도 불친절하기로 정평이 난 2편이 절정이었는데, 그야말로 위의 아앗 시퀀스 그대로, 문자도 따지지도 않고 채집용 3편, 4편에는 승룡이 트이게도 채집 중 낮은 확률로 좋은 아이템을 얻을 수 있을 것 같지만... 주변에서 몬스터들의 기척이 느껴진다.는 메시지가 뜨고 이때 운이 좋으면 레어 아이템을 얻을 수 있지만 신 세계수의 미궁 시리즈, 그 이후에 나온 최신작 5편에서는 채집 방식이 한 턴으로 끝나는 구조로 바뀐 덕분에인지 강제 조우로 다시 회귀해버렸다.... 그나마 위협감지 역활과 같은 버그성 난점들은 수정된 듯하다.

세계수 시스템을 기반으로 한 퍼스나 시리즈와의 클리어 작품 퍼스나 0 세대우 오브 더 레버런스에도 물론 등장. 3, 4편과 같이 파워 스킵에서 채집 도중 메시지가 뜨고, 실패하면 파티에 참가하고 여담으로, 2편에서 채집 도중 아앗이 볼 확률은 고작 1%였다고 한다. 낮아보이는 확률이어도 플레이 중 한 번이라도 일어나는 것을 경험하는 채감 확률을 고려해서 확률을 설정한다고.

그 기원은 1인칭 던전 크롤러의 원조 워저드러에서 합점을 건드렸을 때 나오는 대사 Oope라고 한다.

5) 데이터 정제 결과

	데이터 정제 전	데이터 정제 후
문서의 양(개수)	931029	655452
파일 크기	7,422,121KB	1,491,224KB

丑 1)

b. 검색엔진

부산대학교 인공지능 연구실에서 지원하는 색인 기반 검색엔진이다. 3개의 검색 부 컴포넌트와 하나의 검색관리를 위한 컴포넌트를 포함한다. 검색엔진에 포함되는 컴포넌트는 아래와 같다.

1. 색인어 정렬 / 통합 컴포넌트(Indexing Term Sorting / Merging Component)
2. 역파일 생성 컴포넌트(Inverted Index Generator)
3. 검색 결과 반환 시스템(Retrieval System)
4. 역파일 관리자 / 검색 컴포넌트 래핑(Wrapping) 라이브러리 컴포넌트

또한, 부산대학교 인공지능 연구실에서 지원하는 미리내 검색엔진을 위한 색인기이다. 대상 문서를 형태소 분석하고 그 분석 결과를 바탕으로 색인어를 추출 및 색인한다. 한국어 문장 분석에 필요한 사전을 포함하고 있고, 미리내 색인기에 의해 색인할 문서는 색인처리에 용이 하도록 특정한 형식(ptxt)으로 변환해 주어야한다. ptxt문서의 예는 아래와 같다.

```
<s>
<i> 10 </i>
<u> ./10.ptxt</u>
<t> 테스트 문서 </t>
<c>
홍길동의 나이는 25살이다.
</c>
<e>
```

c. LSTM(단락 선별기)

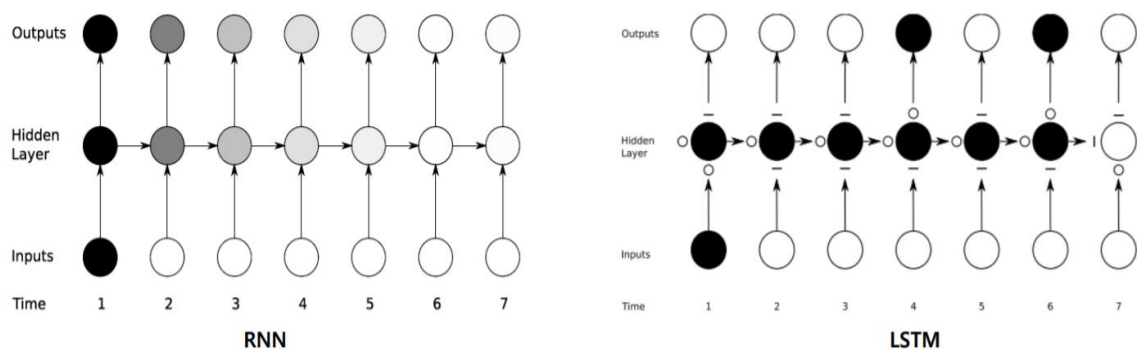


그림 2)

RNN -> 시간이 지나면 이전 입력값을 잊어버린다.

LSTM -> 이전 입력값의 정보가 계속 다음 상태 메모리에 반영되기 때문에 시간이 지나도 이전 입력값을 잊어버리지 않는다.

본 과제에서는, 단락의 길이가 일정하지 않기 때문에 LSTM을 사용하였다. 또한 성능을 유지하면서 처리 시간을 단축시키기 위해 배치 사이즈가 큰 LSTM을 단락선별기로 사용하였다.

1. LSTM 학습

LSTM에 KorQuad를 학습하기 위해 정답이 있는 KorQuad단락과 위키피디아의 임의단락을 구분하여 KorQuad 단락은 학습 Label을 '1', 임의단락은 학습 Label '0'을 주어 학습한다.

2. LSTM 입출력

학습 한 후, 테스트 과정에서 검색엔진에서 추출된 연관 문서 약 100개의 문서를 효율적으로 처리하기 위해 LSTM 모델을 이용해 단락 단위로 변환한다. 하나의 문서당 약 6~9개의 단락이 있기 때문에 600~900개의 단락으로 구성되어진다.

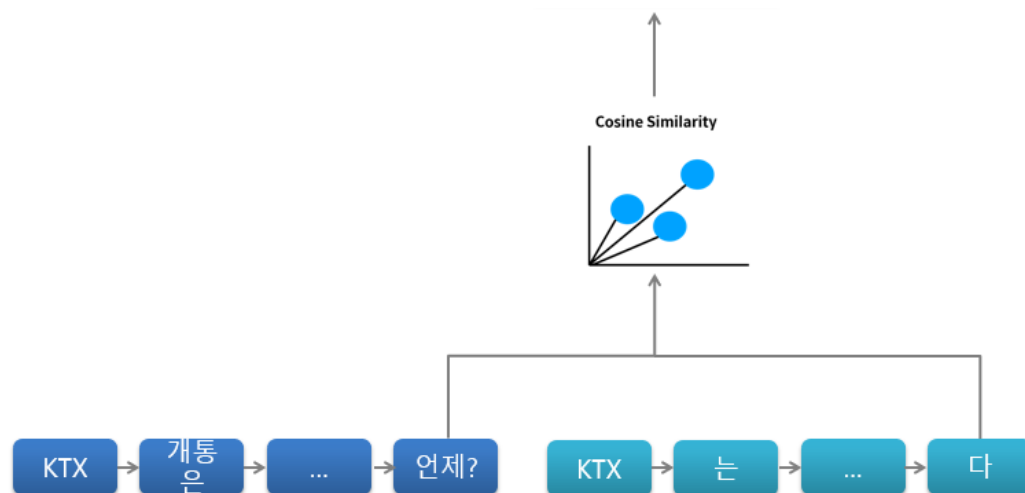


그림 3)

이렇게 선별된 6~900개의 단락 중 연관성이 높은 20개의 단락을 선별한다. 이 때 코사인 유사도를 이용하여 20개의 단락을 선별한다.

d. BERT

자연어 처리에서 사람의 언어를 컴퓨터에서 사용하기 위해서는 단어를 컴퓨터에서 사용할 수 있도록 Vector화하는 과정이 필요하기에 Word Embedding 방식을 사용한다. Word Embedding은 Word2Vec나 Glove 등과 같은 방식을 통해 Language Model을 학습시키고 그것을 통해 얻은 Vector를 단어를 나타내는 Input으로 사용하는 방식이다. 본 과제에서는 GloVe로 학습된 Pre-Trained Model을 이용한다.

본 과제에선 BiDAF, Bert, Google QaNet 모델 중 본 과제에서는 Bert를 사용한다.

```

sentences = [
    '나 는 너 를 사랑 하다 여',
    '나 는 너 를 사랑 하다 였 다',
    '사랑 누 가 말하다 였 나',
]
(q_length, q_tokens, q_embedding, q_ids) = bc.encode(sentences)

love_1 = q_embedding[0][5]
love_2 = q_embedding[1][5]
love_3 = q_embedding[2][1]

spatial.distance.cdist([love_2, love_3], [love_1], metric='cosine')
--
array([[0.0546998 ],
       [0.52740145]])

```

그림 4)

BERT의 가장 큰 특징은 다이나믹 임베딩이라는 점이다. 문장 형태와 위치에 따라 동일한 단어도 다른 임베딩을 갖게 되어 이를 통해 중의성을 해소할 수 있다. 예를 들어 "bank account"와 "bank of the river"의 bank는 Word2Vec 또는 GloVe에서는 동일한 벡터를 갖는다.

그러나 이 단어는 문맥에 따라 전혀 다른 의미를 지녀야 하며 실제로 BERT에서는 전혀 다른 벡터를 갖는다. 모두 동일하게 '사랑' 위치에 대한 워드 임베딩 값을 취했고 코사인 디스턴스를 출력했다. 이 중 문장이 비슷한 1번과 2번의 '사랑' 벡터는 매우 가까운 거리로 나타난다. 반면, 문장이 전혀 다른 3번은 동일한 '사랑'이라도 거리가 상당히 멀다.

1. Bert 입력

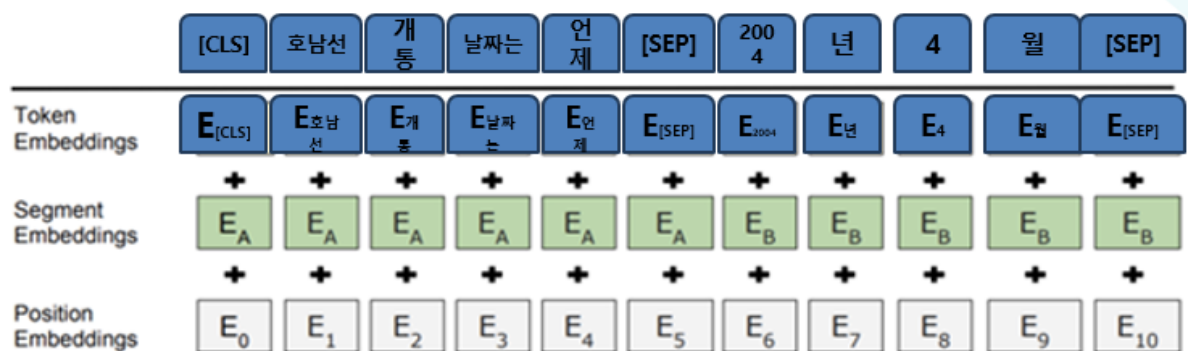


그림 5)

Bert는 위의 그림과 같이 3가지 embedding 방식을 적용하여 임베딩 값을 합하여 input 값에 전달한다. 문장을 토큰화 시키는 Token Embedding, 토큰 A와 토큰 B를 구분하기 위해 이용하는 Segment Embedding, 각 토큰의 위치정보를 주기위한 Position Embedding이 있다. 본 과제에서는 질문과 단락으로 이루어진 Set을 이용한다.

입력 예시 : <CLS> <질문> <SEP> <지문 및 단락> <SEP>

2. Bert 출력

두 개의 Prediction layer를 구성하여 start, end prediction 값의 가중치를 계산하여 질의에 대한 정답을 추출한다.

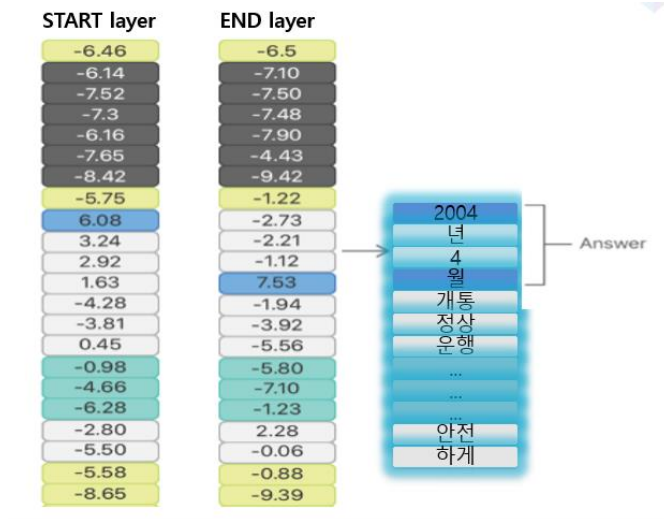


그림 6)

위의 그림은 Softmax하기 전 그림이다. 위의 그림과 같이 start layer에서 가장 큰 확률값과 end layer에서 가장 큰 확률값을 각각 저장한다. 그리고 start와 end 사이의 substring을 출력한다. 그 출력한 값이 정답이다.

"홍길동의 나이는 몇살인가?"

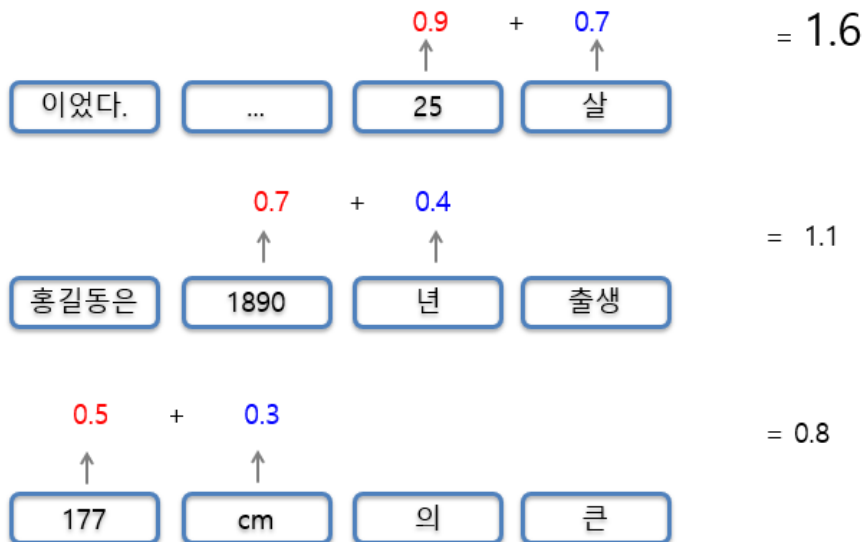


그림 7)

본 과제에서 사용한 방법은 20개의 선별된 단락 중 start, end 확률을 각각 더하여 그 확률값이 가장 큰 substring이 정답이 된다.

e) Django

디장고는 파이썬으로 만들어진 오픈 소스 웹 어플리케이션프레임워크이다. 본 과제에서는 카카오톡 챗봇을 UI로 사용하기 위해 디장고를 사용하였다.

4. 실험환경 및 데이터셋

1. 실험환경

a) 학습데이터

KorQuad 2.0 traning_set(83486개)

KorQuad 2.0 dev_set(10165개)

➔ 최대길이(length400)이하 data set을 선별하여 약 42000개 사용

b) Learning Rate 및 학습횟수

3e-5, 13700회(epoch=2)

c) 개발환경

RTX2080TI(11GB), RAM 32GB

2. 데이터셋

데이터셋 이름	사용 될 모듈	데이터 구성	데이터크기
위키피디아	검색엔진	위키피디아 문서 제목 및 내용	1320446개 문장
KorQuad	딥러닝 자연어 처리 모듈	질문 – 질문의 주제에 해당하는 위키피디아 문서 제목 – 해당 위키피디 아 문서내에서 정답에 해당하는 단어	10,645건의 문단 66,181개의 질의응답
나무위키	검색엔진	나무위키 문서 제목 및 내용	931,029 문서

표 2)

5. 실험결과

1. 검색엔진

상위문서 개수	문서 검색 결과
5	47.9
10	54.0
20	61.3
50	71.7
100	79.6

표 3)

2. LSTM(단락 선별기)

단락 선별 결과		
상위 20	상위 40	상위 80
50.81	57.90	63.58

표 4)

한 문서당 8단락이라 하였을 때 표 2)의 5문서는 40단락이다. 이 때 성능은 47.9이다. 그러나 표 3)의 상위 40단락일 때의 성능은 57.90이다. 이를 보았을 때, 단락 선별기를 이용하였을 때 성능이 더 향상되었음을 알 수 있다.

3. Bert – KorQuad 성능

F1	EM
46.9	30.8

표 5)

- EM : 실제 정답과 예측치가 정확하게 일치하는 비율.
- F1 : 실제 정답과 예측치의 겹치는 부분을 고려한 점수로, EM보다 완화된 평가 척도.

6. 산업체 멘토링 내용 반영 결과

착수 보고서 멘토링 내용	전반적으로 난이도가 높은 프로젝트라 보여집니다. 검색엔진과 질의응답 모델 두 가지를 동시에 구현하는 것은 무리가 있을 것 같다고 생각합니다. 제시된 시스템의 전체적인 정확성은 검색엔진과 질의응답모델 둘 중 하나라도 문제가 있다면 떨어지게 됩니다. 특히 검색엔진이 문서를 제대로 랭킹해주지 않는다면 질의응답모델이 아무리 좋아도 전체 시스템의 유용성이 떨어질 수 밖에 없습니다. 딥러닝에 초점을 맞추고자 한다면 검색엔진 부분은 새롭게 구현하는 것보다 기존의 검색엔진을 활용하는 방법으로 하고 질의응답모델 구현에 집중하는 것이 좋을 것 같다고 생각합니다.
중간 보고서 멘토링 내용	여러가지 실험을 하고 있는 부분은 긍정적이나 실험결과를 기술할 때 Baseline model과 새로운 model에 대한 구체적인 설명을 덧붙여 주시기 바랍니다. 그리고 실험결과 F1 score가 아주 높은 편인데 evaluation에 사용된 데이터셋을 좀 더 자세히 설명해 주시기 바랍니다.
반영 결과	검색엔진과 질의응답모델 두가지를 동시에 구현하는 것은 시간 관계상 무리가 있을 거 같아 검색엔진은 연구실에서 제공하는 미리내 검색엔진을 사용하였다. 이렇게 질의응답 모델인 Bert에 집중하였으나 시간 관계상 Baseline Model을 잡지 못하였다. 하지만 멘토링 의견을 반영하여 실험의 evaluation에 사용된 데이터셋의 설명을 추가하였다.

표 6)

7. 어려웠던 점 및 결론

a. 어려웠던 점

- 1) learning rate, dropout, epoch의 값에 따라 결과의 차이가 크고 직관적으로 예상이지 않아 의미없는 시도를 반복하였다.
- 2) KorQuad 학습을 위한 실험 환경이 열악하여 오랜 시간이 소모되었다.
- 3) 기계독해모델의 정확도, 검색엔진성능을 보아 약 54% 정확도를 기대하였으나, 실제 실험결과는 46.9%로 약 7%의 오차가 있었다. 이는 검색된 문서에 정답이 있음에도 정답이 아닌 부분을 출력하였다.

b. 결론

Bert 출력 부분에서 start와 end의 확률을 단순히 더한 값이 아닌 <CLS>를 이용하거나, 다른 방법을 사용하여 정답을 추출했더라면 보다 더 나은 정답률을 낼 수 있을거라는 결론을 지었다.