스마트 홈 제어를 위한 질의 응답 시스템

흐르므느 조

작성자

201524517 윤태완

201524599 최우성

목차

데이터 생성

■ 질의 분석

- TV 관련 질의 분석 예제

기기	종류	입력	Target Device	Target Info	추출할 정보
TV	전원 여부	TV 켜져 있어	TV	#전원	없음
	예약 프로그램	예약한 프로 조회		#예약	날짜, 시간 정보, 프로그램 정보
	편성표 검색	뉴스가 몇시에 하지		#편성표	날짜, 시간 정보, 프로그램 정보

■ 데이터 생성

초기

- 주먹구구식으로 그때 그때 생각 나는 질의를 데이터에 추가
- 다양한 문장을 만드는 과정에서 시간을 많이 소모
- 효율성이 떨어져 많은 데이터 확보가 어려움

패턴화

- 질의를 만드는데 사용하는 문장을 분석해서 패턴대로 문장을 만들도록 고안함
- 패턴을 만들면 비교적 적은 시간과 노력으로도 많은 데이터 확보 가능

패턴화의 어려움

- 한국어는 조사/어미를 다양하게 활용한 비슷한 문법이 많다.
  - 특정 조사/어미에 사용할 수 있는 동사가 다름.
- 같은 의미라도 그 표현법이 매우 다양함.
  ex) 어제 나는 집에 갔다, 지난날에 나는 집으로 향했었다.

전이 학습에 사용할 모델 찾기

■ BERT 기반 한국어 모델

한국에 Public하게 공개되어 있는 대표적인 한국어 PLM(Pretrained Language Model)의 종류는 SKT의 KoBERT, TwoBlock AI의 HanBERT, ETRI의 KorBERT가 있다.

단점

- KoBERT : Vocab size (8002개)가 상대적으로 작음
- HanBERT : Tokenizer로 인해 리눅스 환경에서만 사용 가능
- KorBERT : API 신청후 사용가능, Tokenizer를 OpenAPI 형태로 제공하여 대량의 데이터를 처리하는 것이 제한됨
- 공통적인 문제 : 사용하려면 tokenizer 파일을 따로 만들어야 하는 단점이 있다.

■ KoELECTRA

이러한 단점을 극복하면서 BERT 이후에 나온 ELECTRA모델을 한국어를 위한 모델로 pre-train한 전이학습 모델이다.

장점

- 필요한 라이브러리를 설치하고 import만 해주면 바로 사용이 가능하다.
- 모든 OS에서 사용가능
- Tokenizer를 따로 생성할 필요가 없다.
- 성능의 보장

● 성능 비교

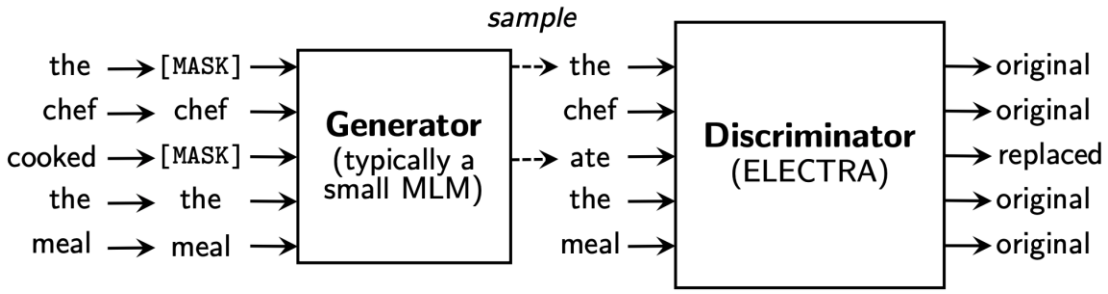
Base Model

	Size	NSMC (acc)	Naver NER (F1)	PAWS (acc)	KorNLI (acc)	KorSTS (spearman)	Question Pair (acc)	KorQuaD (Dev) (EM/F1)
KoBERT	351M	89.63	86.11	80.65	79.00	79.64	93.93	52.81 / 80.27
XLM-Roberta-Base	1.03G	89.49	86.26	82.95	79.92	79.09	93.53	64.70 / 88.94
HanBERT	614M	90.16	**87.31**	82.40	**80.89**	83.33	94.19	78.74 / 92.02
KoELECTRA-Base	423M	**90.21**	86.87	81.90	80.85	83.21	94.20	61.10 / 89.59
KoELECTRA-Base-v2	423M	89.70	87.02	**83.90**	80.61	**84.30**	**94.72**	**84.34 / 92.58**

Small Model

	Size	NSMC (acc)	Naver NER (F1)	PAWS (acc)	KorNLI (acc)	KorSTS (spearman)	Question Pair (acc)	KorQuaD (Dev) (EM/F1)
DistilKoBERT	108M	88.41	84.13	62.55	70.55	73.21	92.48	54.12 / 77.80
KoELECTRA-Small	53M	**88.76**	84.11	74.15	76.27	77.00	93.01	58.13 / 86.82
KoELECTRA-Small-v2	53M	88.64	**85.05**	**74.50**	**76.76**	**78.28**	**93.66**	**81.43 / 90.37**

ELECTRA 모델과 BERT 모델



■ 성능 비교

BERT의 MLM(Masked Language Modeling)

- BERT의 MLM방식은 example에 대해서 15%의 token만 loss로 발생시켜서 학습하기 때문에 비효율적이다. 그리고 학습할 때는 문맥에 MASK token이 존재하지만, 실제 사용시에는 MASK token이 없는 문제가 있다.

ELECTRA의 RTD(Replaced Token Detection) 방식

- ELECTRA의 RTD는 example의 모든 token에 대해서 학습을 하여 BERT보다 훨씬 효율적이고 효과적이다.
  (ELECTRA-Large의 경우 XLNet에 비해 1/4의 계산량으로 비슷한 성능을 볼 수 있다.)

■ 학습 방법

Generator : 작은 MLM(masked language modeling)

- MLM 에서 input 이 주어지면 랜덤한 위치들을 masking하고 masking된 토큰들은 [MASK] 토큰으로 대체한다.
- generator는 masking된 토큰들에 대해 원래 어떤 토큰이 있었을지 예측하는 방법으로 학습한다.

Discriminator

- input token sequence가 original인지 replaced인지 구분하는 이진 분류로 학습한다.

■ 주의점

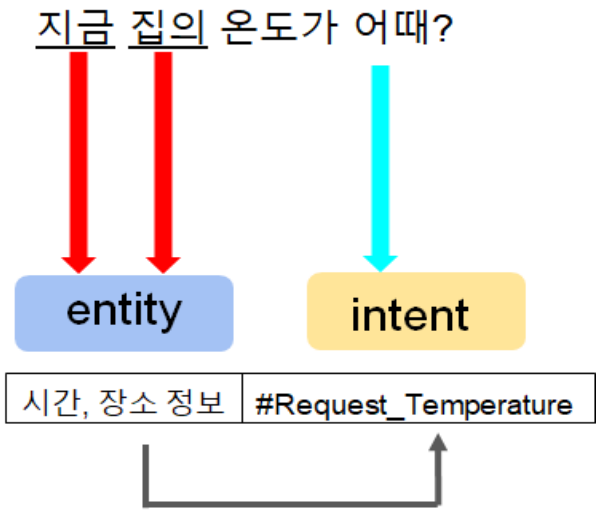
- Generator의 성능이 Discriminator에 비해 유의미하게 성능이 높아지게 되면 Discriminator의 학습에 의미가 없어져서 성능이 오히려 떨어질 가능성이 있다.

■ GAN과 ELECTRA 모델의 차이점

- generator가 원래 토큰과 같은 토큰을 생성하면 GAN에서는 fake 이지만, ELECTRA에서는 positive sample이다.
- GAN은 Generator와 Disciriminator가 서로 적대적으로 학습하지만, ELECTRA는 두 loss의 합을 최소화하도록 학습한다.

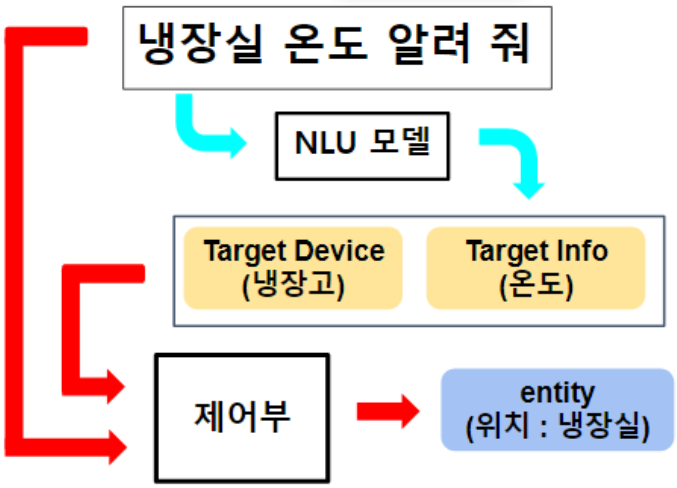
설계 변경 사항

■ 기존 형상



intent와entity를 한번에 찾는다

■ 변경된 사항



설계가 변경된 이유는 entity는 굳이 학습으로 하지 않고 규칙으로 해도 충분할 만큼 패턴이 다양하지는 않았고, Intent는 질의를 분석하면서 한번에 파악하기는 어렵겠다는 판단이 들었다. 왜냐하면 target Device에 대한 Target Info를 알고자하는 것이 intent인데 하나하나가 다 하나의 intent가 될 수 있는 것이었다. 그래서 intent를 target Device와 Target Info에 대한 것으로 각각 나누어 세분화 했다.

실험 및 결과 분석

■ 실험조건

- train set과 test set은 서로 같은 데이터가 없도록 구성하였고, 각 데이터로 사용된 문장은 어미와 조사에 차이를 두었다.

■ 실험목적

- koELECTRA 모델의 사용법을 익히는 것과 모델을 이용해 다수의 label을 구분하는 신경망을 구성하는 것이 가능한지 확인하는 차원에서 진행했다.

■ 실험내용

- TV 전원 상태, 예약, 편성표에 대한 질의 3개 유형에 대한 55000개의

데이터로 학습하고 학습된 것을 train_set으로 평가했다.

■ 실험 결과

- 학습된 모델은 90% 정확도의 성능가지는 것으로 확인되었다.

```
acc_10000 = 0.89936
acc_12000 = 0.89906
acc_14000 = 0.90038
acc_16000 = 0.90132
```

```
값: [ 3.6377652  0.6618045 -9.543202 ]
값: [ 5.419491  -1.5373251 -7.87978  ]
값: [ 0.06971785  3.0246708  -9.74191   ]
값: [-2.4651237  4.498785  -7.186831 ]
```

진행될 계획

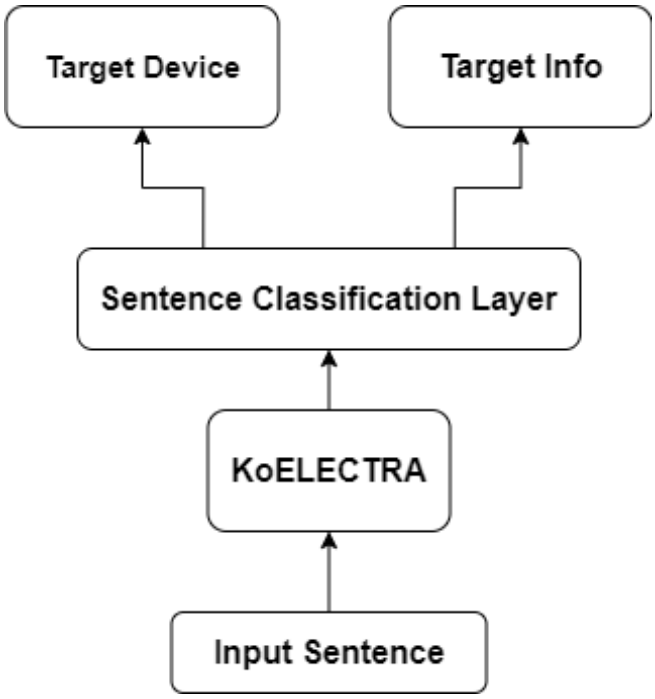
앞으로 진행해야할 계획은 크게 2가지가 있다. 첫번째로 가장 중요한 프로젝트에 사용할 모델을 구축하는 것이다.

두번째로는 모델이 어느정도 만들어지고 진행할 시스템 구축이다.

■ 모델 구축

모델은 MT-DNN으로 만드는 것을 기본으로 하며, 잘 안되는 경우에 각각의 모델을 학습할 것이다.

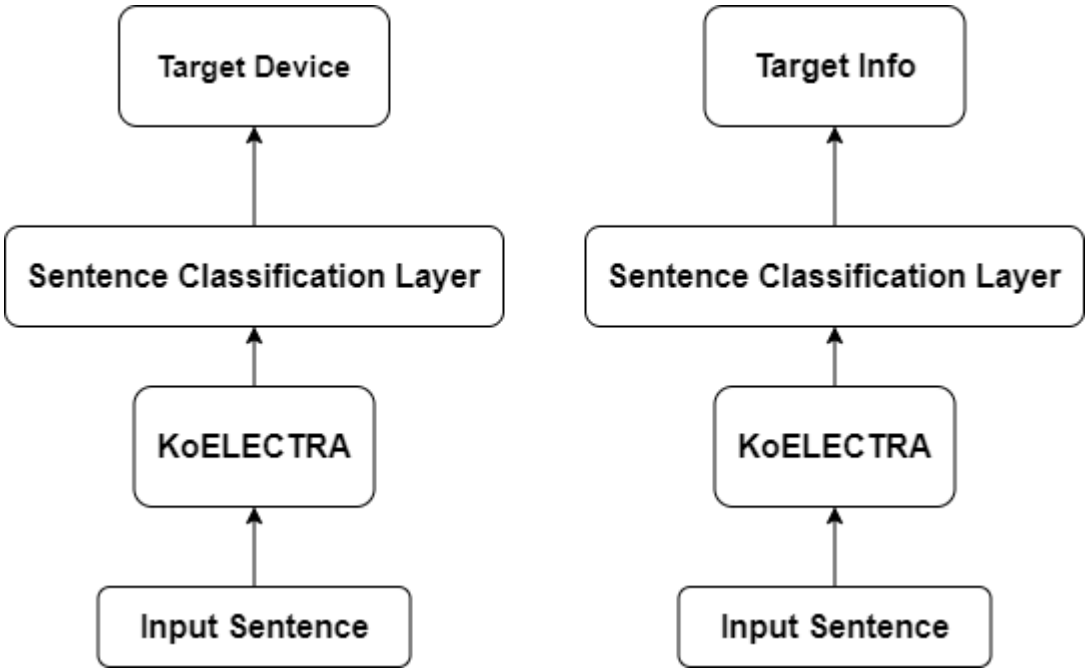
1. MT-DNN



koELECTRA 기반으로 전이학습된 신경망을 기반으로 target device와 Target information을 찾는데 사용한다.

같은 신경망을 공유한다는 특징이 있다.

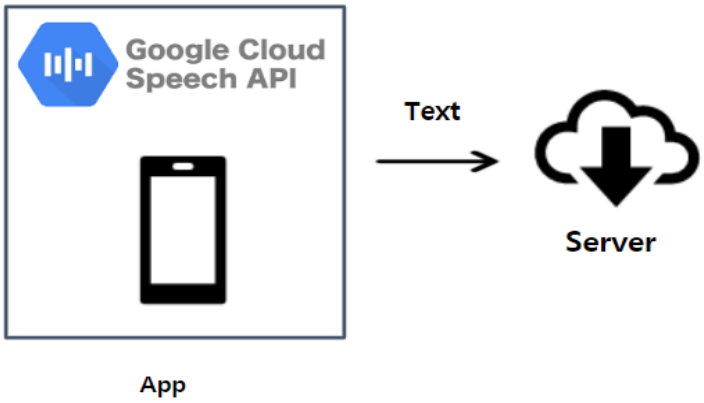
2. 각각의 모델을 사용



차선책으로 각각의 모델을 따로 finetuning해서 모델을 학습시킨 것을 사용하는 것으로 같은 신경망을 사용하지 않는다.

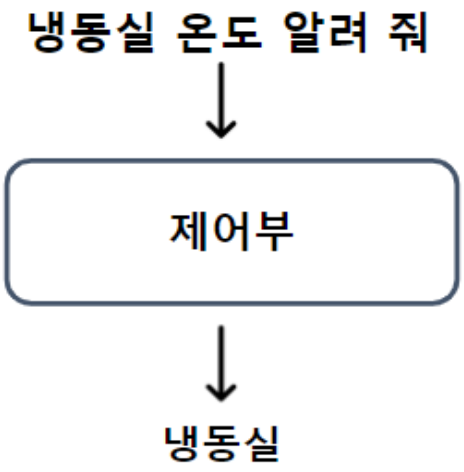
■ 시스템 구축

- App



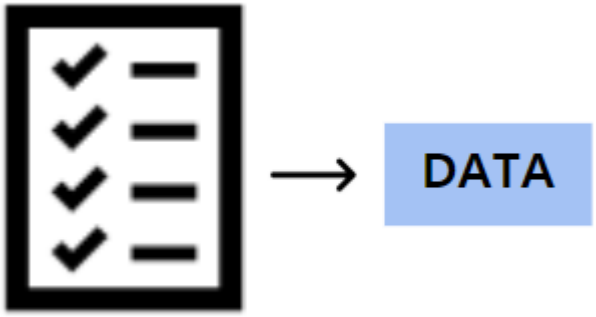
○ 구글의 Speach-To-Text API를 이용하여 음성을 텍스트로 전환함
○ 전환된 텍스트를 서버에 전송함

- 제어부



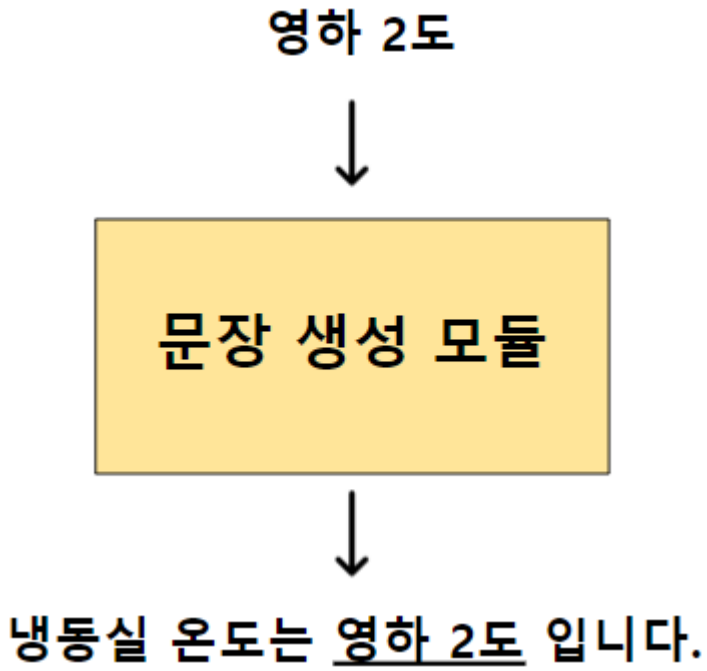
○ entity는 형태소 분석기를 사용해 문장을 분해해서 해당 문장에서 사전에 등록된 단어가 있는지 확인하거나 정규식을 이용해서 잘 분석한다.
○ 분석된 것을 토대로 DB에 쿼리한다

- DB



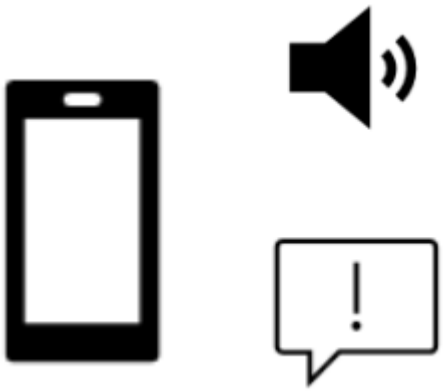
  - 쿼리해야할 것을 잘 구분해 table을 설계한다.

- 문장 생성 모듈



  - ○ 다양한 패턴의 format을 구축
    - ■ ex) "냉동실 온도는" + DATA + "입니다." 형식의 format된 문장으로 돌려준다.

- App



  - 생성된 문장은 내장된 엔진을 사용하여 음성과 문자로 사용자에게 제공한다.

참고한 것들

1. https://monologg.kr/2020/05/02/koelectra-part1/
2. https://github.com/monologg/KoELECTRA/tree/master/finetune
3. https://huggingface.co/transformers/model_doc/electra.html#electraforsequenceclassification