버트를 활용한 스터디룸 예약 챗봇

이선민, 이종희, 노규만, 최종완

1. 목차

- **챗봇 소개 및 데모**
  - ✓ 챗봇의 전망
  - ✓ 스터디룸 예약 챗봇 데모 영상
  - ✓ 챗봇 상용화를 위한 전체 과정 및 사용 툴
  - ✓ 프로젝트 배분 및 팀원 소개
- **왜 버트인가?**
  - ✓ 자연어 처리 모델의 발전 과정
  - ✓ 버트와의 성능 비교
- **버트와 슬롯태깅 소개**
  - ✓ 버트 프리트레인
  - ✓ 버트 파인 튜닝 - 슬롯태깅
- **챗봇 구현**
  - ✓ Pre-train을 위한 사전 준비
  - ✓ Korbert를 이용한 Fine-tuning
  - ✓ 챗봇 훈련 및 평가

2. 챗봇 소개

챗봇 전망

챗봇의 전망

IT/과학>IT/인터넷

'AI 챗봇+보이스봇' AI 컨택센터 '언택트' 붐 타고 급부상, 솔루션 경쟁 가열

채윤정 기자 | 2021-04-18 12:46:11

출처: https://www.metroseoul.co.kr/article/20210418500165

"통계는 이제 코봇에게 물어보세요" 통계청 챗봇 서비스 시작

김덕준 기자 casiopea@busan.com

입력:2021-04-22 21:46:17 수정:2021-04-22 21:46:17 게재:2021-04-22 21:46:33

출처: http://www.busan.com/view/busan/view.php?code=2021042221461734797

HOME > News > 산업

AI상담·챗봇, 공공기관 서비스에도 본격 투입된다…영역 확대중

권상희 기자 | 승인 2020.12.16 17:17 | 댓글 0

출처: http://www.opinionnews.co.kr/news/articleView.html?idxno=44195

미국 시장조사업체 마켓츠앤마켓츠에 따르면 글로벌 챗봇 시장 규모는 2019년 26억 달러(약 2조 8431억원)에서 2024년까지 94억 달러(약 10조2789억원)로 성장할 전망이다. 여기에 AI를 결합한 컨택센터까지 합하면 시장 규모는 더 커질 것으로 추정된다.

> 포지큐브 AI 통합 상담 서비스, 조달청 혁신시제품 시범 사용기관에 선정돼
> 카카오엔터프라이즈, 디지털서비스 전문계약제도 통해 공공IT시장 진출
> 네이버·SK텔레콤도 기술 개발하며 민간에서 사용처 확대중

NUGU

Clova Chatbot Builder

kakaoenterprise CONNECT. SOLVE. CREATE. AI

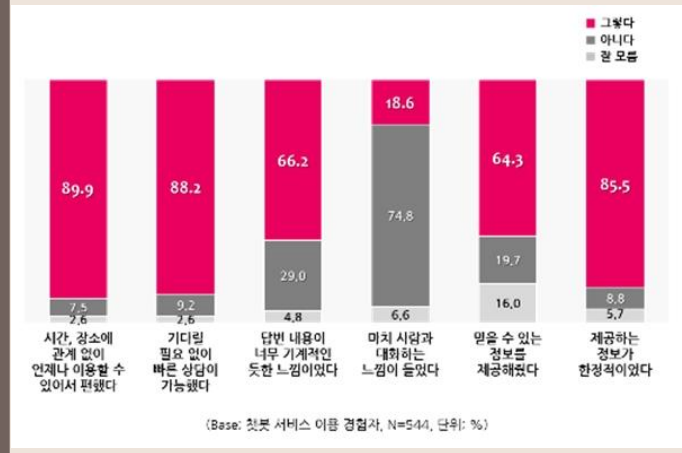
챗봇의 필요성

표 챗봇의 5대 활용 분야

분야	비중
대화형 커머스 및 O2O	쇼핑, 비행기 예약, 숙소 예약, 레스토랑 예약 및 주문, 택시 호출 등
개인비서 서비스	헬스케어, 뉴스피드, 날씨 정보, 금융 상담, 일정 관리, 길 찾기 등
공공 서비스	법률 상담, 세금 납부, 부동산 정보, 구인구직
엔터테인먼트 서비스	광고, 미디어, 방송 안내, 데이팅, 공연 등
기업용 메신저	정보 검색, 파일 공유, 데이터 보관, 팀원 정보 공유, 자동 사무화(OA), CRM

자료: 한국정보화진흥원

챗봇의 한계



출처: https://m.post.naver.com/viewer/postView.nhn?volumeNo=30147412&memberNo=44045763

'챗봇(Chat bot)' 서비스 인지 및 이용 여부

(Base: 전체, N=1,000, 단위: %)

조사 기간 : 2020년 11월 19일~11월 24일

조사 대상 : 디지털기기를 사용하는 만 19세~59세 성인남녀 1,000명

3. 데모 영상

데모 영상
유튜브 링크

4. 전체 과정 및 사용 툴

데이터 수집 및 정제

사전 학습을 위한 데이터

뉴스 기사 웹 크롤링을 통한 말뭉치 확보

정규표현식을 이용해 말뭉치 정제

챗봇 구현을 위한 데이터

스터디룸 업체에 직접 문의해 예약에 꼭 필요한 5가지 정보 파악

멀티턴 대화 구축용 스크립트 작성

(프로그래밍을 통한 데이터 부풀리기로 문장 자동 생성 +직접 수기로 문장 작성)

버트 Pre-training

Vocab 만들기 및 토큰화

구글 Sentencepiece로 어휘 사전을 만들고 말뭉치 토큰화 진행

실제 Pre-training 진행

ALBERT를 이용해 사전 학습을 직접 실시

ETRI에서 KorBERT 다운

미리 학습된 모델을 API Key로 파일 다운로드

버트 Fine-tuning

챗봇 데이터 다듬기

4300여 문장을 6:2:2 비율로 분할

seq.in, seq.out 파일 생성

챗봇 모델 구축

다듬은 데이터로 챗봇 모델 훈련

Fine-tuning 모델 평가

F1 Score로 평가

웹서비스로 상용화

Flask를 이용한 웹 구현

웹에서 채팅을 통해 예약이 가능하도록 연동

Flask_ngrok으로 배포

가볍고 빠른 ngrok으로 배포

예약완료 문자 발송 서비스

예약 완료 메시지 전송

✓ 깃허브 주소: https://github.com/K-107/BERTsDay

6. 왜 버트인가?

자연어 처리 모델의 발전 과정, 버트와의 성능 비교

✓ **버트는 자연어 처리 모델 중 성능이 뛰어나다.**

자연어 처리(natural language processing)란? 인간의 언어를 컴퓨터가 처리할 수 있도록 숫자로 바꾸는 것.

통계 기반 임베딩

TF-IDF, 단어-문맥 행렬

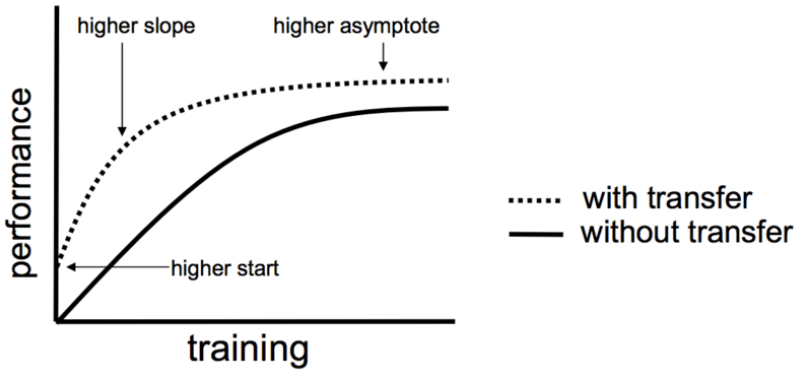
단어 수준 임베딩

NPLM, Word2Vec,
FastText, GloVe, Swivel

단어&문장 수준 임베딩

ELMo, GPT, BERT
(전이 학습 모델,
예측 기반 모델)

BERT(Bidirectional Encoder Representations from Transformer)는 성능이 뛰어난 트랜스포머 블록을 사용하는 마스크 언어 모델로 문장의 빈칸에 해당하는 단어와 다음 문장을 예측하는 과정에서 학습하며 양방향으로 문장을 인식한다.



출처: Handbook Of Research On ML Applications and Trends

System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	Average -
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.9	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	88.1	91.3	45.4	80.0	82.3	56.0	75.2
$BERT_{BASE}$	84.6/83.4	71.2	90.1	93.5	52.1	85.8	88.9	66.4	79.6
$BERT_{LARGE}$	**86.7/85.9**	**72.1**	**91.1**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**81.9**

출처: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

7. 버트와 슬롯태깅 소개

✓ 버트는 BPE를 이용하여 토큰화를 수행한다.

센텐스피스(Sentencepiece)를 이용한 토큰화

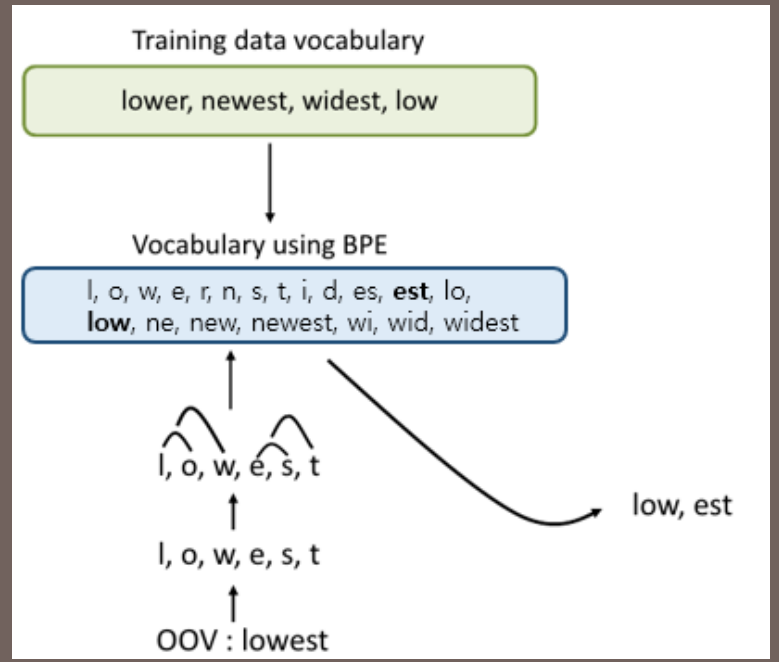
- 센텐스피스(Sentencepiece)는 BPE 알고리즘을 이용하여 토큰화한다.
- BPE(Byte Pair Encoding): OOV(Out-Of-Vocabulary) 문제를 해결하기 위한 서브워드 분리(Subword segmenation) 알고리즘이다.
- 작동 원리는 연속적으로 가장 많이 등장한 글자의 쌍을 찾아서 하나의 글자로 병합한다.

기존 토큰화 방식

```
# dictionary
# 훈련 데이터에 있는 단어와 등장 빈도수
low : 5,
lower : 2,
newest : 6,
widest : 3

# vocabular
low, lower, newest, widest

# 새로운 단어 등장
'lowest' => OOV
```

BPE 알고리즘 방식

```
# dictionary
l o w : 5, l o w e r : 2, n e w e s t : 6, w i d e s t : 3

# vocabulary
l, o, w, e, r, n, w, s, t, i, d

# first dictionary update!
l o w : 5, l o w e r : 2, n e w es t : 6, w i d es t : 3

# first vocabulary update!
l, o, w, e, r, n, w, s, t, i, d, es

# second dictionary update!
l o w : 5, l o w e r : 2, n e w est : 6, w i d est : 3

# second vocabulary update!
l, o, w, e, r, n, w, s, t, i, d, es, est

# final dictionary update!
low : 5, low e r : 2, newest : 6, widest : 3

# vocabulary update!
l, o, w, e, r, n, w, s, t, i, d, es, est, lo, low, ne, new,
newest, wi, wid, widest

# 새로운 단어 등장
'lowest' => 'l, o, w, e, s, t' => 'low', 'est'
```



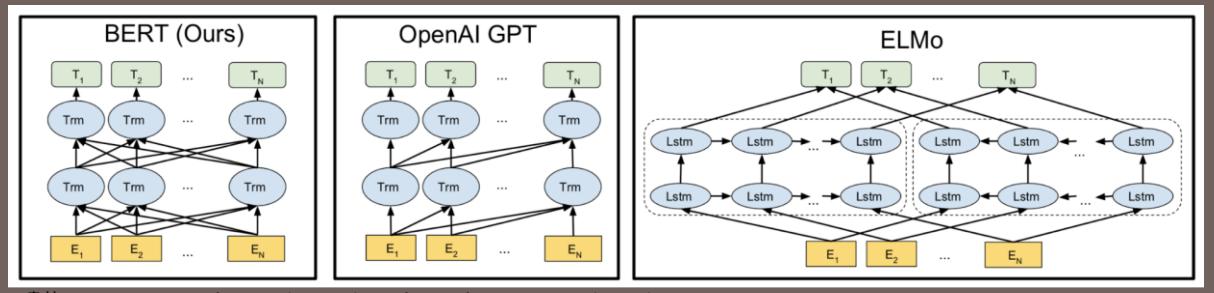
출처: Neural Machine Translation of Rare Words with Subword Units

8. 버트와 슬롯태깅 소개

✓ **버트는 [MASK]된 단어와 두 문장의 연결성을 예측하며 학습한다.**

- 마스크 언어 모델**(masked language model)** : 텍스트 데이터 자체가 입력값이자 정답이 되는 비지도 학습 모델. 입력 문장의 일부 단어들을 마스킹**(MASK)**해서 모델이 스스로 마스킹된 단어가 무엇인지 예측한다. 빈칸만 맞추면 되기 때문에 양방향 언어 모델이다.

✓ 언어 모델(masked language)인 GPT(Generative Pre-Training), ELMo(Embeddings from Language Model)과 BERT의 비교.



출처: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

	입력값
입력값	배우 임윤아 너무 예뻐요
마스킹 후보	배우 **임윤아 너무 예뻐요**
마스킹 후 입력값	배우 [MASK] **오늘** 예뻐요

- 다음 문장 예측**(Next sentence prediction)**: 입력으로 주어진 두 문장이 이어진 문장인지 아닌지를 예측하는 것을 학습

[CLS] 배우 [MASK] 오늘 예뻐요 [SEP] 내일 날씨는 [MASK] 예정입니다 [SEP]

9. 버트와 슬롯태깅 소개

✓ 버트는 문맥을 반영하여 임베딩(Contextual Embedding)을 한다.

- 트랜스포머: seq2seq의 LSTM을 **어텐션**으로 대체해 문장에 대한 정보 추출 성능을 높힌 모델.
- 셀프 어텐션: **Query, Key, Value**가 모든 같은 경우로 문장에서 각 단어끼리 얼마나 관계가 있는지 측정하는 방법.

Ex) 배우 임윤아 아주 예뻐요

문장 내 단어들의 백터값

배우	임윤아	아주	예뻐요
0.53	1.58	-0.45	4.14
2.56	2.15	-0.21	2.45
1.48	-1.15	0.54	1.12
· · ·	· · ·	· · ·	· · ·

어텐션 맵	배우	임윤아	아주	예뻐요
배우	15	24	8	12
임윤아	24	16	9	107
아주	8	9	11	13
예뻐요	12	107	13	12

‘배우’ 벡터

0.53	2.56	1.48	· ·

‘임윤아’ 벡터

1.58	2.15	1.15	· ·

내적

어텐션 스코어

24

어텐션 맵	배우	임윤아	아주	예뻐요
배우	15	24	8	12

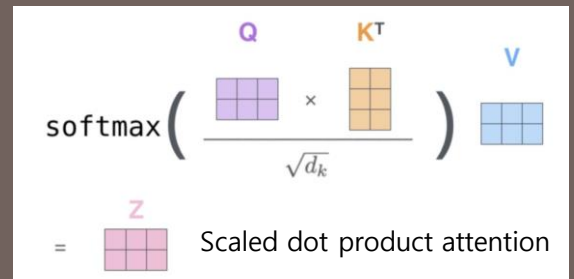
Softmax

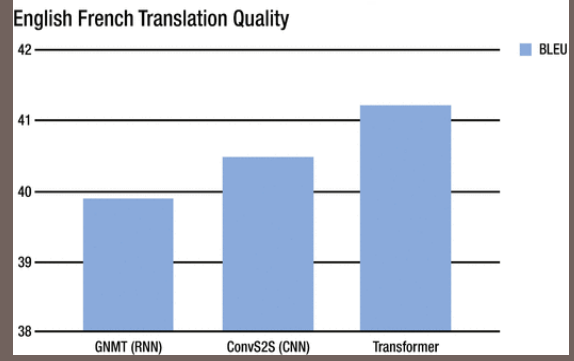
Softmax	배우	임윤아	아주	예뻐요
배우	0.2	0.5	0.15	0.15

스칼라곱

배우	임윤아	아주	예뻐요
0.53	1.58	-0.45	4.14
2.56	2.15	-0.21	2.45
1.48	-1.15	0.54	1.12
· · ·	· · ·	· · ·	· · ·

배우	임윤아	아주	예뻐요
0.104	0.79	-0.067	0.621
0.512	1.075	-0.031	0.367
0.296	-0.575	0.081	0.169
· · ·	· · ·	· · ·	· · ·





출처: Attention Is All You Need

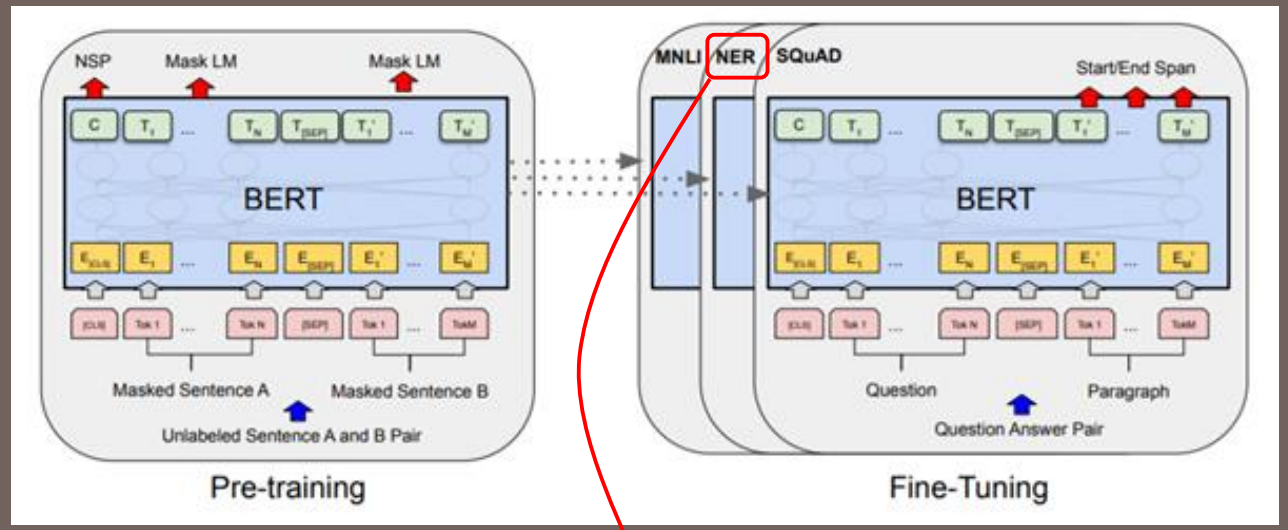
‘배우’에 대한 문맥 백터
1.448
1.923
-0.029
· · ·

10. 버트와 슬롯태깅 소개

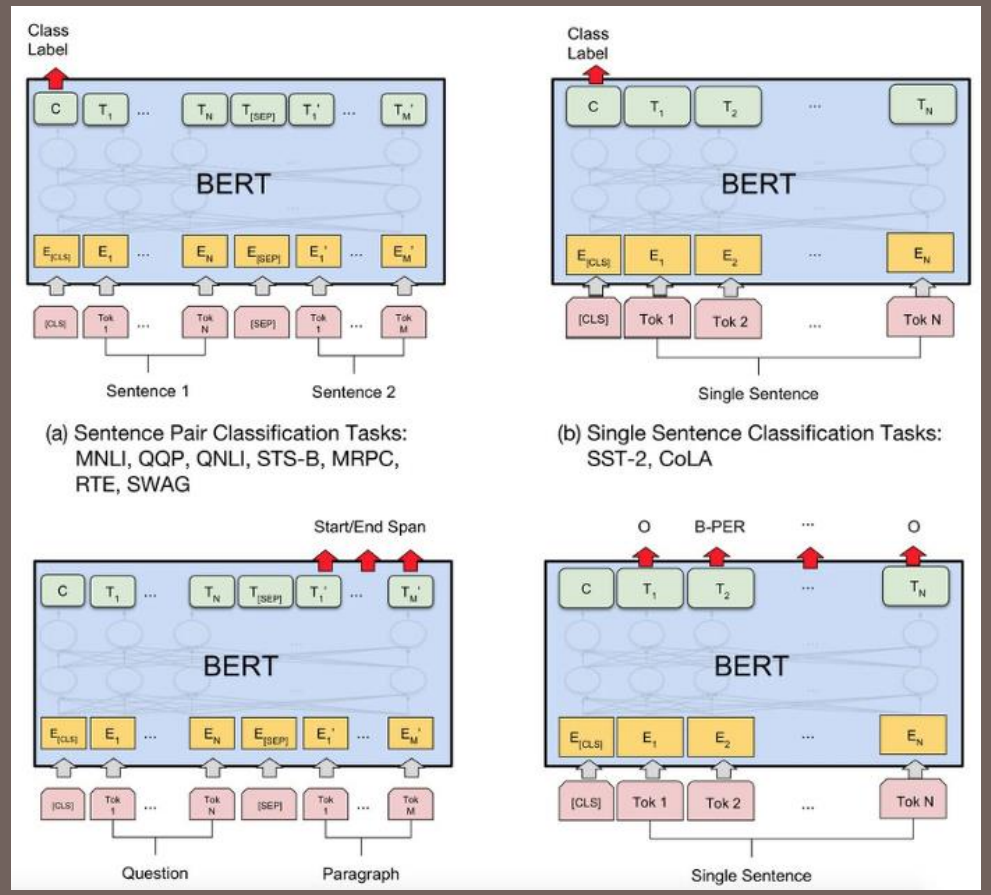
✓ 버트는 전이학습을 통해 문제를 해결한다.

파인 튜닝: 프리트레인 이후 추가 학습을 시행해 임베딩을 다운스트림 태스크에 맞게 업데이트하는 것.

- 사전 학습된 가중치를 가지고 있는 프리트레인 버트 모델을 입력값으로 사용하여 다양한 하위 문제에 미세 조정하여 사용(전이 학습)할 수 있다.



Named Entity Recognition



출처: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

11. 버트와 슬롯태깅 소개

- 슬롯태깅: 버트의 미세조정(Fine-tuning)에서 이루어지는 과정으로, 문장에서 의미적인 개념 **(Semantic concepts)**을 추출하는 역할을 한다.

임윤아	엑시트	보	고	싶	어

↓

BERT pretrained model (ETRI KorBERT)

↓

임윤아	엑시트	보	고	싶	어
1.43	2.54	0.23	3.42	1.65	0.12
3.45	5.43	1.24	1.23	0.43	1.94
. . .	. . .	. . .	. . .	. . .	. . .

↓

Fine-tuning layer (Dense layer = Fully connected layer = ANN)

↓

	임윤아	엑시트	보	고	싶	어
배우 이름	0.8	0.2	0.1	0.1	0.1	0
영화 제목	0.2	0.7	0.2	0	0.1	0.1
슬롯 아님	0	0.1	0.7	0.9	0.8	0.9
(확률) 합계	1	1	1	1	1	1

✓ **Pre-train model의 아웃풋을 Fine-tuning layer의 인풋으로 사용**

✓ **Fine-tuning시 슬롯을 넣은 문장틀을 학습시킴.**

✓ **Fine-tuning layer의 아웃풋**

12. 버트와 슬롯태깅 소개

입력 (토큰 수, N) 행렬

임윤아	1.43	3.45	· · ·
엑시트	2.54	5.43	· · ·
보	0.23	1.24	· · ·
고	3.42	1.23	· · ·
싶	1.65	0.43	· · ·
어	0.12	1.94	· · ·

X
내적

Fine-tuning layer (N, 슬롯 수) 행렬

배우 이름	영화 제목	슬롯 아님
0	0.22	0.4
0.42	0.43	0
· · ·	· · ·	· · ·

=

(토큰 수, 슬롯 수) 행렬

	배우 이름	영화 제목	슬롯 아님
임윤아	1.7	0.4	0.1
엑시트	0.5	1.5	0.3
보	0.1	0.2	2.4
고	0.1	0.3	1.1
싶	0.2	0.1	1.5
어	0.3	0.1	1.6

Softmax

(토큰 수, 슬롯 수) 행렬

	배우 이름	영화 제목	슬롯 아님
임윤아	0.8	0.2	0
엑시트	0.2	0.7	0.1
보	0.1	0.2	0.7
고	0.1	0	0.9
싶	0.1	0.1	0.8
어	0	0.1	0.9

13. 챗봇 구현

프리트레인

BERT's DAY K-107 BIG DATA

1. 프리트레인 데이터 생성

다음 뉴스기사 웹크롤링 후 정규표현식으로 정제

output2.txt

2. 코랩에 알버트 클론

버트 모델은 크기가 커서 메모리가 부족과 학습 시간이 너무 오래 걸리는 문제가 있다. **버트의 라이트 버전인 알버트**는 크기를 획기적으로 줄였으며 오히려 성능은 높혔다.

Model		Parameters	Layers	Hidden	Embedding	Parameter-sharing
BERT	base	108M	12	768	768	False
	large	334M	24	1024	1024	False
	xlarge	1270M	24	2048	2048	False
ALBERT	base	12M	12	768	128	True
	large	18M	24	1024	128	True
	xlarge	59M	24	2048	128	True
	xxlarge	233M	12	4096	128	True

Table 2: The configurations of the main BERT and ALBERT models analyzed in this paper.

Model		Parameters	SQuAD1.1	SQuAD2.0	MNLI	SST-2	RACE	Avg	Speedup
BERT	base	108M	90.5/83.3	80.3/77.3	84.1	91.7	68.3	82.1	17.7x
	large	334M	92.4/85.8	83.9/80.8	85.8	92.2	73.8	85.1	3.8x
	xlarge	1270M	86.3/77.9	73.8/70.5	80.5	87.8	39.7	76.7	1.0
ALBERT	base	12M	89.3/82.1	79.1/76.1	81.9	89.4	63.5	80.1	21.1x
	large	18M	90.9/84.1	82.1/79.0	83.8	90.6	68.4	82.4	6.5x
	xlarge	59M	93.0/86.5	85.9/83.1	85.4	91.9	73.9	85.5	2.4x
	xxlarge	233M	**94.1/88.3**	**88.1/85.1**	**88.0**	**95.2**	**82.3**	**88.7**	1.2x

3. 단어 사전 생성

정제된 txt 데이터를 Sentencepiece를 이용하여 vocab (단어사전)을 만듦

```
import sentencepiece as spm
train = --unk_piece=[UNK] # (미등록 단어 토큰)
        --pad_piece=[PAD] # (최대 길이를 맞추는 용도)
        --bos_id=-1 eos_id=-1 # (begin, end of sentence token id)
        --pad_id=0 # (pad tokoen id)
        --unk_id=1 # (unknown token id)
        --user_defined_symbols=[CLS],[SEP],[MASK] # (문장 시작, 종료, 마스크 토큰)
        --input={큰 텍스트파일 경로} # (학습시킬 파일.txt)
        --model_prefix={보캡 파일 이름} # (만들어질 모델 이름)
        --model_type=bpe # (사용할 모델, BPE 알고리즘 모델)
        --vocab_size={단어 집합의 크기}
spm.SentencePieceTrainer.Train(train)
```

vocab_sample.model
vocab_sample.vocab

4. 텍스트 파일 분리

vocab을 만들 때 사용한 큰 txt 파일을 직접 토큰화하지 않고, tfrecord을 이용하여 여러 파일로 작게 쪼개서 사용한다.

```
# 토큰화
! for file in `ls /content/drive/MyDrive/mi/data | grep small_`; #
do python -m albert.create_pretraining_data #
--input_file=/content/drive/MyDrive/mi/data/$file #
--output_file=/content/drive/MyDrive/mi/data/${file}.tfrecord #
--vocab_file=/content/drive/MyDrive/mi/data/vocab_sample.vocab #
--spm_model_file=/content/drive/MyDrive/mi/data/vocab_sample.model; done
```

small_00
small_00.tfrecord
small_01
small_01.tfrecord

5. 알버트 훈련

albert_config.json

```
{
  "attention_probs_dropout_prob": 0, (드롭 아웃 비율)
  "hidden_act": "gelu", (Pointwise Feedforward Networks의 활성함수 종류, 아래서 설명)
  "hidden_dropout_prob": 0,
  "embedding_size": 128,
  "hidden_size": 768, (트랜스포머 블록 입출력 행렬의 차원 수)
  "initializer_range": 0.02, (모델 가중치 초기화 범위, 분산을 의미)
  "intermediate_size": 3072, (Pointwise Feedforward Networks의 중간 레이어 벡터 차원 수)
  "max_position_embeddings": 512, (토큰 순서 정보를 유지할 토큰 수)
  "num_attention_heads": 12, (어텐션 헤드 수)
  "num_hidden_layers": 12, (트랜스포머 블록 수)
  "num_hidden_groups": 1,
  "net_structure_type": 0,
  "gap_size": 0,
  "num_memory_blocks": 0,
  "inner_group_num": 1,
  "down_scale_factor": 1,
  "type_vocab_size": 2, (세그먼트 임베딩의 종류 수, 2로 고정)
  "vocab_size": 5000 (어휘 집합의 크기, '4. vocab 만들기'의 vocab_size와 일치해야함)
}
```

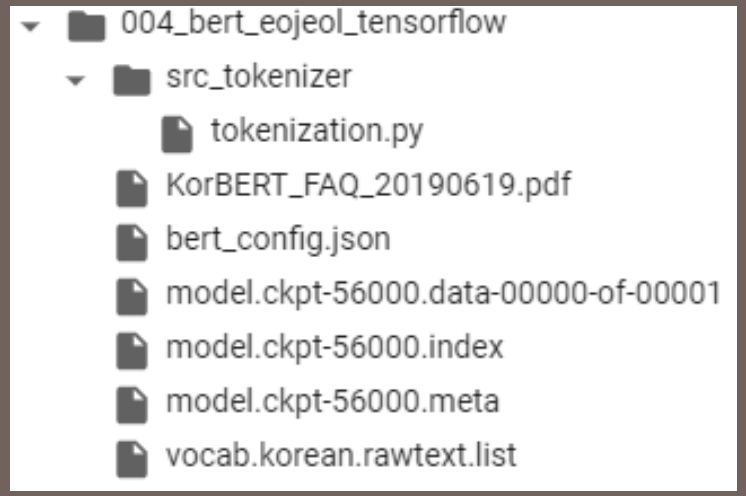
```
# 프리트레인 실행
! python -m albert.run_pretraining #
--input_file=/content/drive/MyDrive/mi/data/small_00.tfrecord, #
/content/drive/MyDrive/mi/data/small_01.tfrecord #
--output_dir=/content/drive/MyDrive/mi/data #
--albert_config_file=/content/drive/MyDrive/mi/albert_config.json #
--do_train --do_eval --train_batch_size=10 --eval_batch_size=10 #
--max_seq_length=512 --max_predictions_per_seq=20 --optimizer='lamb' #
--learning_rate=.00176 --num_train_steps=10000 #
--num_warmup_steps=1000 --save_checkpoints_steps=1000
```

model.ckpt-9000.data-00000-of-00001
model.ckpt-9000.index
model.ckpt-9000.meta
model.ckpt-best.data-00000-of-00001
model.ckpt-best.index
model.ckpt-best.meta

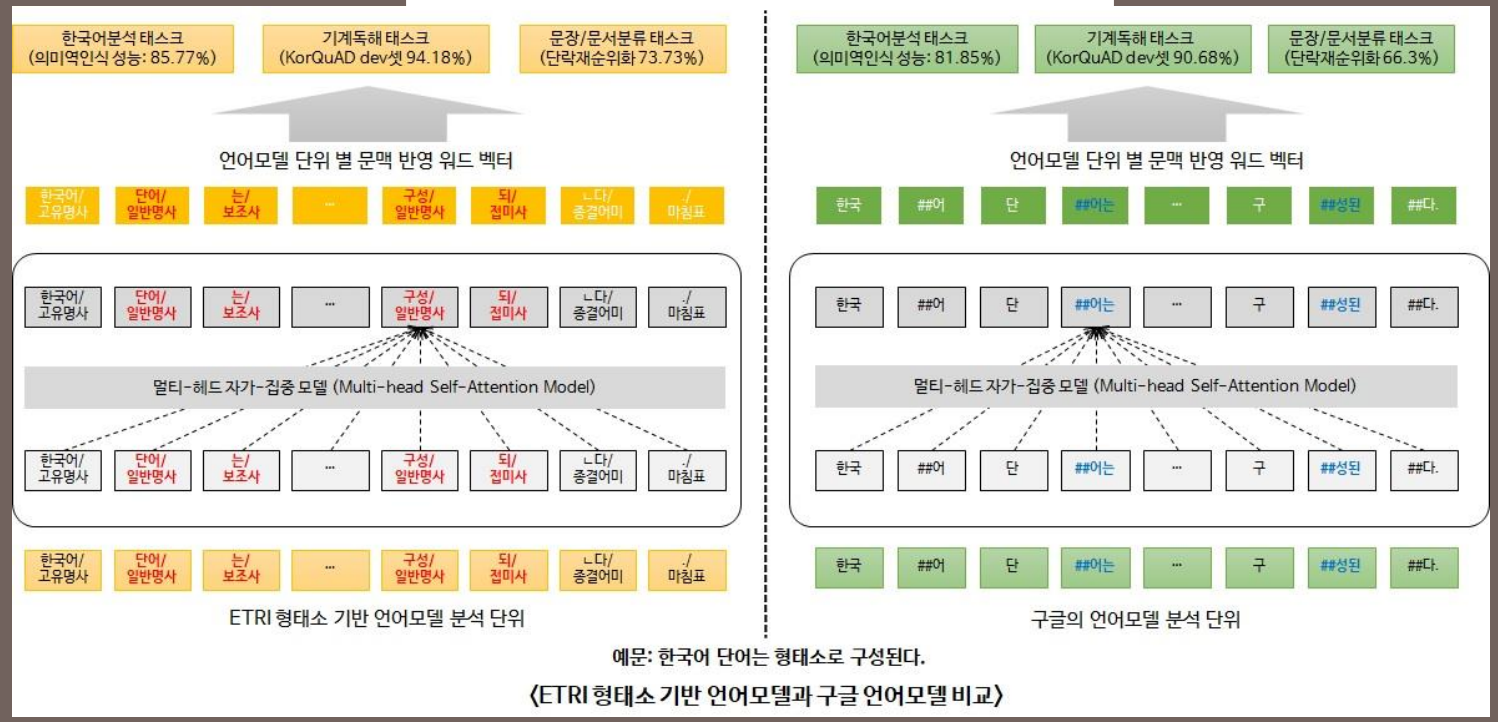
14. 챗봇 구현

KorBERT 모델

- 최소 단어 사전은 **최소 3만 정도**가 필요하지만 컴퓨팅의 한계로 프리트레인 모델은 만들 수 없었음.
- ETRI에서 제공하는 **한국어 BERT 언어모델**을 Pre-train 모델로 선정.
- 구글이 배포한 한국어 언어모델과 비교 평가한 결과, ETRI의 언어모델이 평균 **4.5% 성능이 우수**한 것으로 평가
- 한국어 언어모델 학습 말뭉치로는 신문기사와 백과사전 등 **23GB**의 대용량 텍스트를 대상으로 학습



한국어에서 KorBERT의 우수성



〈ETRI 형태소 기반 언어모델과 구글 언어모델 비교〉

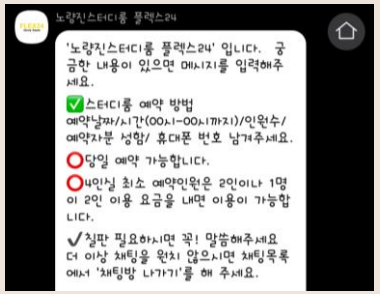
배포 모델	세부 모델	세부 내용	모델 파라미터
KorBERT	Korean_BERT_WordPiece	▪ 학습데이터: 23GB 원시 말뭉치 (47억개 형태소) ▪ 딥러닝 라이브러리: pytorch, tensorflow ▪ 소스코드: tokenizer ▪ Latin alphabets: Cased	30797 vocabs, 12 layer, 768 hidden, 12 heads,

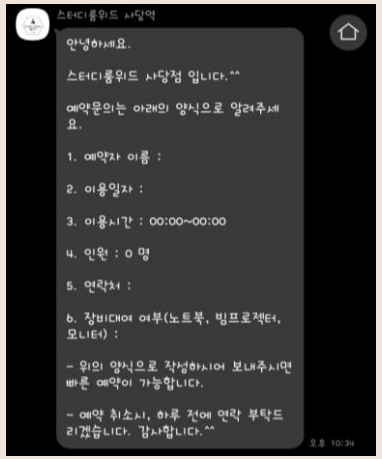
15. 챗봇 구현

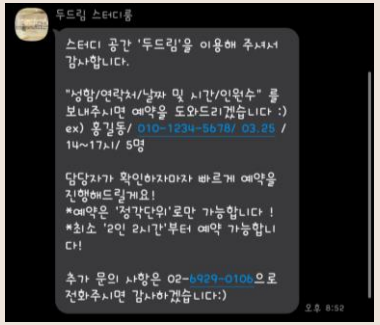
파인 튜닝 데이터 생성

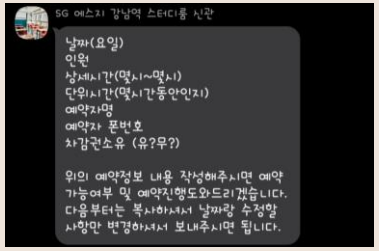
슬롯 선정

- 실제 스터디룸 업체들에 직접 문의하여 얻은 대화 자료를 토대로 필요한 슬롯들을 선정.









총 5가지 슬롯들:
<이름(3글자)> <번호(000-0000-0000)>
<인원(명)> <날짜(00월, 00일)>
<시간 (00시{시작 & 종료})>

슬롯을 넣은 문장 생성

- 챗봇을 위한 버트 파인 튜닝 데이터를 생성.
- 코드로 데이터 부풀리기를 통한 3000여 문장 생성
- 직접 수기로 작성한 1000여 문장 생성
- 총 **4300여 문장**으로 파인 튜닝 학습 데이터로 사용

Ex) 사장님 사람이 /인원;4명/인데 예약되나요? 번호는 /번호;010-1887-9298/이고 날짜가 /날짜;3월 3일/ 입니다. 제 이름은 /이름;남다정/이며 예약 시간은 오후 /시작시간;13시/부터 /종료시간;16시/까지에요.

코드로 생성한 문장들

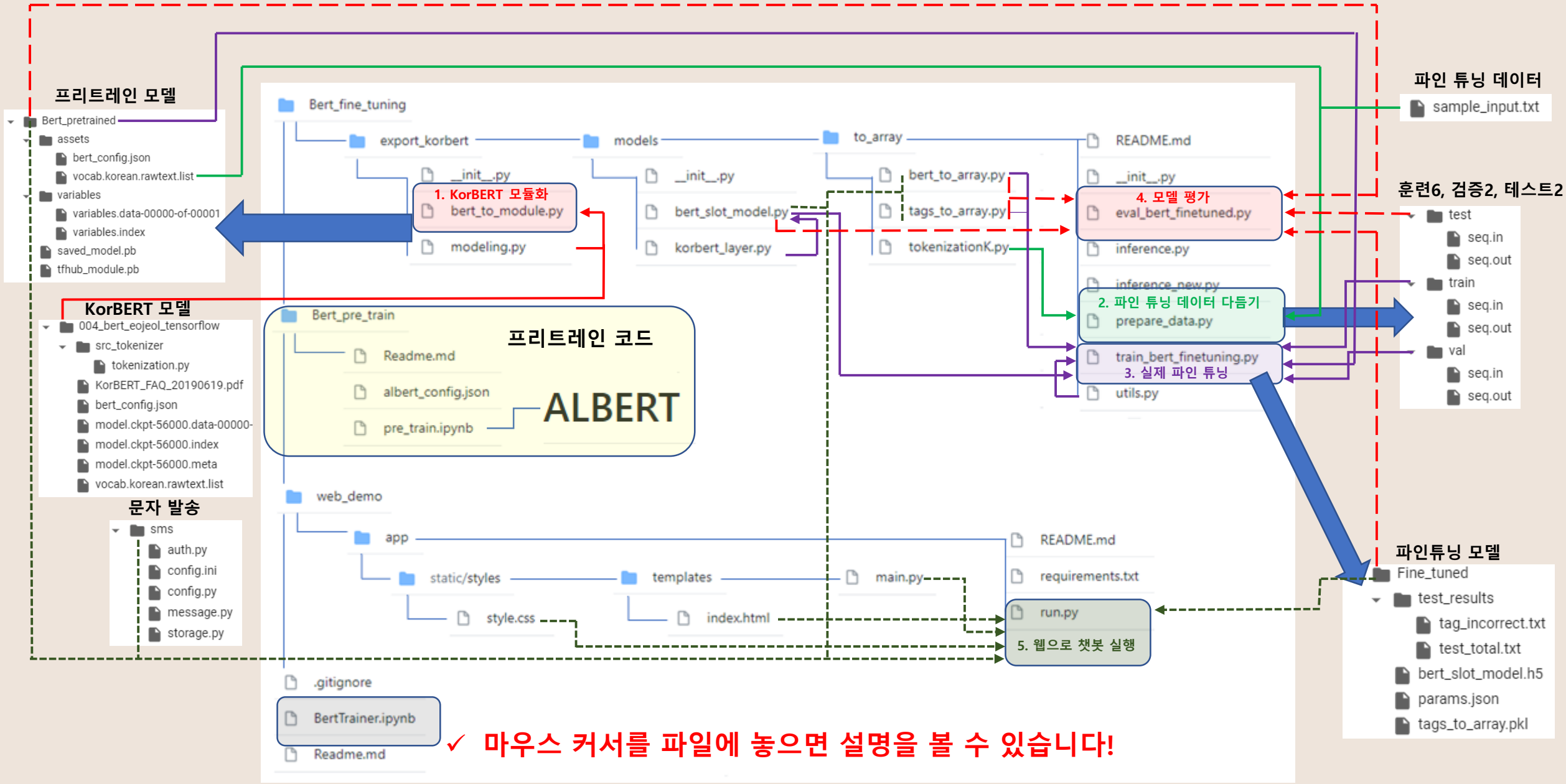
안녕하세요 총 인원 /인원;6명/ 이름 /이름;고시우/ /시작시간;9시/부터 /종료시간;12시/까지 날짜는 /날짜;4월 21일/ /번호;010-3405-8096/
안녕하세요 총 인원 /인원;6명/ 이름 /이름;권준우/ /날짜;6월 21일/에 /번호;010-9521-9452/ 번호로 /시작시간;7시/에 빈 방 예약 되나요
예약하겠습니다 /날짜;1월 1일/ 이름 /이름;장서연/ 번호 /번호;010-3405-8096/으로 /인원;8명/ 부탁드립니다
예약하겠습니다 /날짜;1월 20일/ /인원;9명/ /시작시간;7시/부터 /이름;오서윤/ 부탁드립니다

수기로 작성한 문장들

스터디룸 예약하고싶은데 날짜가 /날짜;5월 11일/에 괜찮을까요?
안녕하세요 저 스터디룸 사용하고싶어서 연락드리는데 /날짜;6월 17일/에 예약하고싶어요.
친구들이랑 공부하려고 하는데 /날짜;7월 19일/에 스터디룸 예약될까요?
사장님 예약할게요. 날짜가 /날짜;8월 14일/입니다.

16. 챗봇 구현

파인 튜닝을 위한 깃허브 파일 구조



✓ 마우스 커서를 파일에 놓으면 설명을 볼 수 있습니다!

17. 챗봇 구현

성능 평가

- 슬롯이 아닌 경우 'O'로 표시되는데 슬롯 'O'에 대한 비중이 다른 슬롯들에 비해 훨씬 크기 때문에 정확도(Accuracy)로는 평가가 불가하다.
- 재현율(Recall)과 정밀도(Precision)의 조화평균인 **F1 Score**를 이용한다.

		실제	
		Positive	**Negative**
예측	**Positive**	**True Positive (TP)**	**False Negative (FN)**
	Negative	**False Positive (FP)**	**True Negative (TN)**

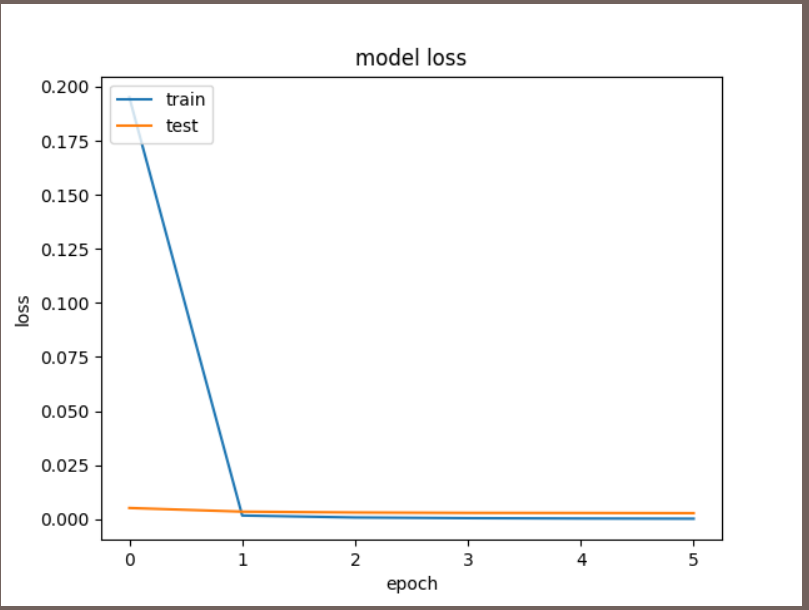
슬롯 예측에 대한 F1 Score 값

```
Slot f1_score = 0.9997689143596616
```

오인식한 슬롯들을 포함한 문장들

```
sent ['[CLS]', '5월_', '9일_', ',_', '5', '명_', ',_', '13', '시_', '에_', '사용', '하고_', '싶', '습니다', '._', '[SEP]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]',
'[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]',
'[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]',
'[PAD]', '[PAD]', '[PAD]']
pred: ['날짜', '날짜', 'O', '인원', '인원', 'O', '종료시간', '종료시간', 'O', 'O', 'O', 'O', 'O', 'O']
score: [0.99976104, 0.99938416, 0.9996611, 0.9995747, 0.9997203, 0.9996562, 0.9353863, 0.72465396, 0.9999875, 0.9999939, 0.99999523, 0.99999833, 0.9999974, 0.99998236]
ansr: ['날짜', '날짜', 'O', '인원', '인원', 'O', '시작시간', '시작시간', 'O', 'O', 'O', 'O', 'O', 'O']

sent ['[CLS]', '제가_', '제_', '친구', '들_', '7', '명_', '이라', 'ㅇ', '_', '공', '부', '할', '건', '데_', '[SEP]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]',
'[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]',
'[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]',
'[PAD]', '[PAD]']
pred: ['O', 'O', 'O', 'O', '인원', '인원', 'O', '이름', 'O', 'O', 'O', 'O', 'O', 'O']
score: [0.99975353, 0.99788696, 0.98099685, 0.9865145, 0.99967015, 0.99909484, 0.6400223, 0.38590682, 0.9450895, 0.9985568, 0.99977726, 0.9999789, 0.9999962, 0.9999944]
ansr: ['O', 'O', 'O', 'O', '인원', '인원', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O']

sent ['[CLS]', '안', '녕', '하', '세요_', '컨', '퍼', '런', '스로_', '대', '관_', '하려고_', '하는데', '요', '._', '8', '명_', '이', '서_', '해야_', '합니다', '._', '이름', '은_', '박',
'물', '관', '이고_', '로_', '연락', '주', '세요_', '[SEP]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]',
'[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]']
pred: ['O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', '인원', '인원', 'O', 'O', 'O', 'O', 'O', 'O', 'O', '이름', '이름', '이름', 'O', 'O', 'O', 'O', 'O']
score: [0.9999975, 0.999997, 0.9999976, 0.99999666, 0.96036154, 0.94691837, 0.953348, 0.99941134, 0.99962056, 0.99959475, 0.9999975, 0.99999857, 0.999998, 0.99999416,
0.9996325, 0.99884534, 0.99977714, 0.9999149, 0.99999774, 0.99999857, 0.9999989, 0.9999845, 0.999974, 0.9804326, 0.9719911, 0.97605395, 0.9999403, 0.99997246, 0.9999975,
0.9999962, 0.999998]
ansr: ['O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', 'O', '인원', '인원', 'O', 'O', 'O', 'O', 'O', 'O', 'O', '이름', '이름', '이름', '이름', 'O', 'O', 'O', 'O']

sent ['[CLS]', '네_', 'ㅈ', 'y', 'p', '_', '입니다', '._', '[SEP]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]',
'[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]',
'[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]', '[PAD]',
'[PAD]', '[PAD]', '[PAD]', '[PAD]']
pred: ['O', '이름', '이름', '이름', 'O', 'O', 'O']
score: [0.98448795, 0.8473706, 0.89749813, 0.63344383, 0.64690816, 0.99988616, 0.99993646]
ansr: ['O', '이름', '이름', '이름', '이름', 'O', 'O']
```

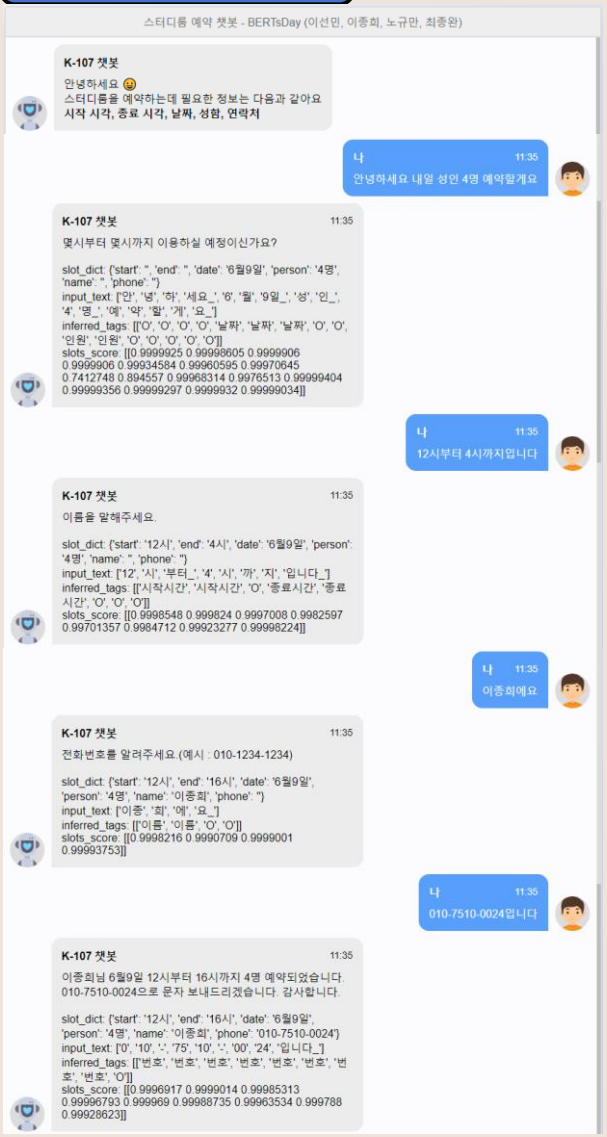
Epoch 5까지 훈련, 테스트 데이터 손실률



18. 챗봇 구현

입력 문장의 슬롯 예측와 출력 문장

문장 분석 결과



출력 문장 생성

- 버트는 디코더가 없으므로 문장을 생성할 수 없다.
- 출력 문장은 하이브리드(룰베이스드 + 머신러닝) 방식으로 생성된다.

```
answer_name_arr = ['성함이 어떻게 되시나요?',
                   '이름을 말해주세요.',
                   '예약하시는 분 성함이 어떻게 되시나요?',
                   '예약자님 성함도 한번만 알려주시겠어요?']
answer_phone_arr = ['연락 가능한 번호를 써주세요.(예시 : 010-1234-1234)'
                    '전화번호를 알려주세요.(예시 : 010-1234-1234)',
                    '예약자 분의 번호를 입력해주세요.(예시 : 010-1234-12
                    '연락 가능한 번호 하나만 남겨주시겠어요? (000-0000-0
                    '예약하시는 분께 연락할 수 있는 번호를 남겨주시겠어요
answer_date_arr = ['몇 월 며칠에 예약하고 싶으신가요?',
                   '예약하고 싶은 월일을 입력해주세요. (예시: 1월 3일)',
                   '예약하시려는 날짜를 알려주세요.',
                   '방문하시려는 날짜는 언제인가요?']
```

```
try:
    # 1. 사용자가 입력한 한 문장을 슬롯태깅 모델에 넣어서 결과 뽑아내기
    for i in range(0,len(inferred_tags[0])):
        if slots_score[0][i] >= app.score_limit:
            if inferred_tags[0][i]=='날짜':
                input_date += token_list[i]
                app.slot_dict['date'] = re.sub('_','',input_date)

            elif inferred_tags[0][i]=='시작시간':
                input_start += token_list[i]
                if app.question != "end":
                    app.slot_dict['start'] = re.sub('_','',input_start)
                else:
                    app.slot_dict['end'] = re.sub('_','',input_start)
```

참고 자료:

유원준(2021),『딥 러닝을 이용한 자연어 처리 입문』, 위키독스
이기창(2019),『한국어 임베딩』, 에이콘
전창욱, 최태균, 조중현, 신성진(2019),『텐서플로 2와 머신러닝으로 시작하는 자연어 처리』, 위키북스
사이토 고키(2019),『밑바닥부터 시작하는 딥러닝2』, 한빛미디어

Qian Chen, Zhu Zhuo, Wen Wang(2019), BERT for Joint Intent Classification and Slot Filling
Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, Radu Soricut(2019), ALBERT: A Lite BERT for Self-supervised Learning of Language Representations
Ashish Vaswani Noam Shazeer Niki Parmar Jakob Uszkoreit Llion Jones Aidan N. Gomez Lukasz Kaiser Illia Polosukhin(2017), Attention is All You Need

https://zsunn.tistory.com/?page=5
https://lovit.github.io/nlp/2018/04/02/wpm/
https://keep-steady.tistory.com/37
https://ratsgo.github.io/nlpbook/docs/tokenization/encode/#bert
https://goodtogreate.tistory.com/entry/Saving-and-Restoring
https://lv99.tistory.com/17

미디어젠 연구원들의 교육자료