

# 2023.03.08 미팅 자료

Sentiment Classification, chatbot, bert-as-service

팀원 : 배한성, 김은서, 조현아, 조유진

# CONTENTS

1. Sentiment Classification
2. chatbot
3. bert-as-service

# Sentiment Classification

A dark blue, abstract, curved shape that starts from the bottom left and sweeps upwards and to the right, filling the bottom half of the slide.

# kobert + nsmc - training

- kobert 모델 + 네이버 영화 감성리뷰(nsmc) 데이터셋 학습 (리뷰데이터 15만개)
- train batch size : 32 , eval batch size : 64 , epoch : 5
- 정확도 89%, loss 0.43

```
Iteration: 100% 4688/4688 [27:00<00:00, 2.89it/s]
Epoch: 100% 5/5 [2:17:31<00:00, 1650.23s/it]
03/06/2023 04:42:21 - INFO - trainer - ***** Model Loaded *****
03/06/2023 04:42:21 - INFO - trainer - ***** Running evaluation on test dataset *****
03/06/2023 04:42:21 - INFO - trainer - Num examples = 50000
03/06/2023 04:42:21 - INFO - trainer - Batch size = 64
Evaluating: 100% 782/782 [02:24<00:00, 5.43it/s]
03/06/2023 04:44:45 - INFO - trainer - ***** Eval results *****
03/06/2023 04:44:45 - INFO - trainer - acc = 0.89198
03/06/2023 04:44:45 - INFO - trainer - loss = 0.4374107635553802
```

---

# kobert + nsmc - predict

- 임의로 크롤링 데이터에서 무작위로 100개의 별점 1점의 부정적인 문장 추출하여 예측진행
- 100개의 문장 중 긍정으로 잘못 분류한 문장 수는 17개
- 잘못 분류한 문장 분석결과. 부정적인 표현이 없고 부정적인 뉘앙스만 있는 문장은 긍정으로 혹은 중립적인 문장을 긍정으로 예측함

1 처음 제품 받았을 때 모니터에 85라는 문구가 찍혀있어서 재무팅 하면 없어지는 건지  
2 쪽 쪽으로 사진 증명하고 요청 일단 당장 사용하기 위해서 주문했는데 경 하나 찍힌  
3 결함 제품을 보내놓고 왜 소비자가 결함을 증명해야 되는지 이해되지 않고 서비스센터  
4 사용하다 고장 난 게 아니고 애초부터 이런 제품인 경우 판매처에서 반품 처리해서 결  
5 마침 당일에 휴가여서 서비스센터에 다녀올 수 있었는데 만약에 제가 직장 때문에 센터  
6 이라 방에서 판매만 열심히 하지 마시고 사후 처리에 신경 좀 써주세요  
7 그리고 리뷰 절대 삭제하지 마세요  
8 한 달 후 후기 씹. 22년 6월 9일 발은 노트북이 고장 나서 서비스 받았습니다  
9 한 달가량 지었는데 서비스센터에서 키보드 불량이라고 키보드 키판 교체했습니다  
10 서비스센터까지 가야 했고 수리 완성까지 부품이 없어서 다음날 수리되었어요  
11 너무 수고롭네요  
12 키보드 불량 증상 올립니다  
13 씹 산지 얼마나 되었다고 고장일까요  
14 스피드 뽀에서는 일주일만 지나서 교환 환불 안된다고 하네요  
15 규정이 그렇다니 뭐 어쩔 수 없죠  
16 삼성 서비스센터에서는 일 년 안에 세 번 같은 고장이 발생하면 반품 환불 가능하  
17 한 달 전에 쓴 후기 단점 노트북 속도가 너무 느리고 충전이 빨리 닳아져요  
18 주식하려고 샀는데 속도가 느려서 산 걸 정말 후회하고 있어요  
19 전체적으로 속도가 느려요  
20 SSD까지 추가해서 샀는데 속도와 무관했어요  
21 노트북이 커서 무거워요 장점 외관은 화면이 큼니다  
22 썬 게 비지떡이네요  
23 수량 약 일주일째 터치패드 불량품을 받았고 불량품이란 걸 확인받기 위해 생고생을  
24 5일 동안 시간 낭비함  
25 스피드 뽀 고객센터에서 하드웨어 불량일 수 있다는 걸 인지하고 우선 교환 조치했  
26 다른 분들 리뷰 보니 불량률이 어느 정도 있는 거 같은데 이 정도면 먼저 테스트 후  
27 저는 내일 우선 스피드 뽀 고객센터에 연락해보고 후속 조치 취하겠습니다  
28 제발 한 달 리뷰는 좋은 내용으로 작성하면 좋겠습니다  
29 1점도 아깝습니다  
30 노트북은 제외하고 같이 옵션으로 선택한 마우스가 스크롤 내리는 소리가 엄청 크고  
31 그래서 문의했는데 단순 번식은 환불이 안된다고 하시네요  
32 단순 번식이 아니라 이상이 있어서 문의한 거고 그럼 서비스센터를 가라든가 어떤 방  
33 삼성 공식 판매점 CS가 이 정도일 줄은 상상도 못했네요  
34 총네 치킨집보다 못한 CS 수준입니다

67 0  
68 0  
69 0  
70 0  
71 0  
72 0  
73 0  
74 1  
75 0  
76 0  
77 0  
78 0  
79 1  
80 1  
81 0  
82 0  
83 0  
84 0  
85 0  
86 0  
87 0  
88 0  
89 0  
90 0  
91 0  
92 1  
93 1

# bert fine-tuning

- **2023.02.28)** BertForSequenceClassification 모델 fine-tuning 진행
  - BertForSequenceClassification.from\_pretrained("bert-base-multilingual-cased", num\_labels = 2)
  - 정확도 = 89%
  - 하지만, 긍/부정 예측 결과가 만족스럽지 않았음
    - train data의 문제라고 예상

=> 테스트1) 1점, 5점 리뷰만 사용

테스트2) 테스트1 조건 + 데이터 추가

# bert fine-tuning

- 테스트1

- 데이터 : 1, 5점 리뷰 (총 31,607개)

- 부정 : 10,782개 (34.11%)

- 긍정 : 20,825개 (65.89%)

- batch\_size = 16

- epochs = 5

=> 결과) accuracy = 0.92

- 실제 predict 결과

- kobert + nsmc와 동일 데이터 사용

- 100개 중 88개 부정으로 판단

```
===== Epoch 4 / 5 =====
Training...
Batch 500 of 1,423. Elapsed: 0:02:23.
Batch 1,000 of 1,423. Elapsed: 0:04:47.

Average training loss: 0.13
Training epoch took: 0:06:49

Running Validation...
Accuracy: 0.91
Validation took: 0:00:13

===== Epoch 5 / 5 =====
Training...
Batch 500 of 1,423. Elapsed: 0:02:23.
Batch 1,000 of 1,423. Elapsed: 0:04:47.

Average training loss: 0.10
Training epoch took: 0:06:48

Running Validation...
Accuracy: 0.92
Validation took: 0:00:13

Training complete!
```

# bert fine-tuning

- 테스트2
    - 데이터 : 1, 5점 리뷰 (총 78,858개)
      - 부정 : 33,049개 (41.91%)
      - 긍정 : 45,809개 (58.09%)
    - batch\_size = 16
    - epochs = 10
- => 결과) accuracy = 0.88
- 실제 predict 결과
    - kobert + nsmc와 동일 데이터 사용
    - 100개 중 93개 부정으로 판단

```
Batch 3,000 of 3,549. Elapsed: 0:14:27.  
Batch 3,500 of 3,549. Elapsed: 0:16:51.
```

```
Average training loss: 0.08  
Training epoch took: 0:17:05
```

```
Running Validation...
```

```
Accuracy: 0.88  
Validation took: 0:00:32
```

```
===== Epoch 10 / 10 =====
```

```
Training...
```

```
Batch 500 of 3,549. Elapsed: 0:02:24.  
Batch 1,000 of 3,549. Elapsed: 0:04:49.  
Batch 1,500 of 3,549. Elapsed: 0:07:13.  
Batch 2,000 of 3,549. Elapsed: 0:09:38.  
Batch 2,500 of 3,549. Elapsed: 0:12:02.  
Batch 3,000 of 3,549. Elapsed: 0:14:27.  
Batch 3,500 of 3,549. Elapsed: 0:16:51.
```

```
Average training loss: 0.06  
Training epoch took: 0:17:06
```

```
Running Validation...
```

```
Accuracy: 0.88  
Validation took: 0:00:32
```

```
Training complete!
```



chatbot



# 현재 지원하는 챗봇 서비스들

- CLOVA Chatbot

- 네이버에서 제공하는 챗봇 서비스
- 다국어 지원
- 자연어 처리 기술과 머신러닝 기반 학습 알고리즘 (엔진 성능 지속적으로 향상)
- 대화 모델 빌드 10회 및 1,000건의 API 호출 매달 제공
- 호출할 수 있는 모델 내부 메소드 없다.

- chatGPT

- OpenAi에서 제공하는 챗봇 서비스
- gpt-3.5-turbo 사용
- \$0.002/1K tokens (1000 tokens = 750 words)
- fine-tuning gpt-3.5-turbo X => fine-tuning base GPT-3 models O
- 모델 학습 시키는데도 비용이 든다.
- [Chat completion - OpenAI API](#)

# 현재 지원하는 챗봇 서비스들

- koGPT

- 카카오브레인에서 출시한 gpt-3 기반 한국어 언어 생성모델
- REST API로 월간 제공량 최대 1,000건으로 사용 제한 => 이후부터는 유료
- 텍스트를 분류, 검색, 요약 또는 생성하는데 가장 적합한 모델
- 몇 개의 단어를 입력하면 여러 개의 문장으로 구성된 문단을 완성시켜 준다.
- 따라서, 본 프로젝트의 질문, 답변 기능에 적합X
- 호출할 수 있는 모델 내부 메소드가 없다.

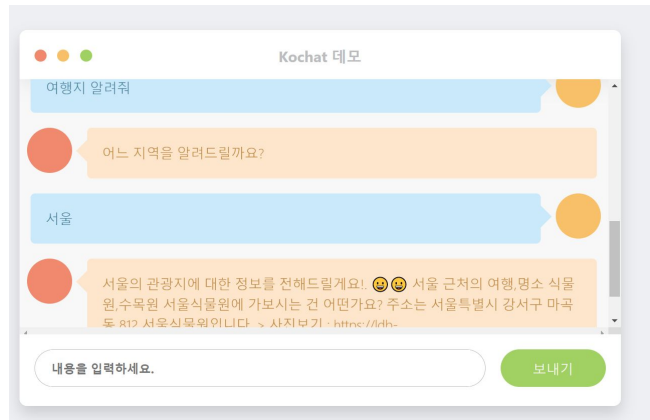
- KoGPT2

- SKT-AI에서 한국어성능 한계 개선을 위해 개발한 모델
- GPT-2 모델을 Fine-Tuning한 한국어 언어모델
- pre-trained KoGPT2를 이용한 chatbot 실험이 존재
- 호출할 수 있는 내부 메소드 없다.
- 하지만 우리가 원하는 키워드를 문장에 포함시켜 output으로 만들 수 있지 않다.

# 현재 지원하는 챗봇 서비스들

- Kochat

- 한국어를 지원하는 최초의 오픈소스 딥러닝 챗봇 프레임워크
- 모델
  - embed 모델 - Gensim의 Word2Vec/FastText 모델 사용
    - embed 모델로써 Gensim만 지원 (BERT 지원 X)
  - intent 모델 - Bidirectional LSTM / 커스텀 (torch로 구현)
  - entity 모델 - Bidirectional LSTM / 커스텀 (torch로 구현)
- BIOES 태깅을 적용한 데이터와 OOD 데이터 이용
- 시나리오
  - 어떤 intent에서 어떤 entity가 필요하고 어떤 api를 호출하는지 정의
- KochatApi
  - Flask로 구현
  - RESTful API 제공
- [Kochat - Opensource Korean chatbot framework](#)



bert-as-service



# bert-as-service란

- BERT as Service는 BERT를 사용하여 Sentence Embedding을 얻을 수 있다.
- BERT as Service는 ZeroMQ를 통해 단, 두줄의 코드로 문장을 고정 길이 벡터로 매핑할 수 있다.
  - ZeroMQ는 비동기 메시징 라이브러리이다.
  - ZeroMQ는 메시지 큐를 제공하지만 전용 메시지 브로커 없이 동작이 가능하다.
- BERT as Service는 12/24 Layer의 BERT 모델을 기반으로 한다.

# bert-as-service란

- 문장 벡터의 크기
  - 각 문장은 768차원의 벡터로 변환된다.
  - 단, fine-tuning된 BERT의 pooling층에 따라 다를 수 있다.
- Embedding Vector
  - 임베딩 벡터를 얻을 때 기본적으로 마지막에서 두 번째 Layer의 output을 평균 Pooling하여 얻는다.
  - 마지막에서 두 번째 Layer를 사용하는 이유
    - 마지막 Layer는 사전 훈련 중 학습 방법(마스킹된 단어 예측 등)에 따라 값이 편향될 수 있다.
    - 단, 마지막 레이어는 파인 튜닝에 따라 사용 가능할 수 있다.
    - 첫번째 Layer를 사용하면 원래 단어 정보를 보존할 수 있다.(셀프 어텐션을 사용하지 않은 값)
    - 따라서 첫번째 ~ 마지막 레이어의 선택은 트레이드 오프이다.

# bert-as-service란

- BERT as Service가 동시에 처리할 수 있는 요청 수
  - 최대 동시 요청 수는 **Server**를 생성할 때 결정할 수 있다.
  - 요청 수가 최대 요청 수 보다 많으면 대기 큐에서 기다렸다가 작업을 수행한다.
  - 요청 수는 문장의 수가 아닌 클라이언트에서 보낸 문장 목록을 의미한다.
  - 요청 수는 보유한 **GPU/CPU**의 수보다 적거나 같아야한다.
    - 그렇지 않으면 여러 작업이 하나의 **GPU/CPU**에 할당되어 메모리 부족 현상이 발생할 수 있다.
- 여러개의 **Layer** 사용
  - 서버를 실행할 때 사용할 **Layer**들을 결정할 수 있다.

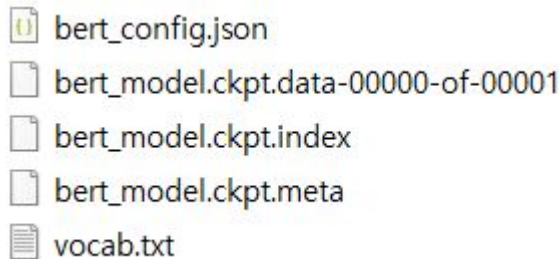


# bert-as-service란

- 사용 가능한 Pooling 방법
  - NONE
    - Pooling이 전혀 없으며 문장 임베딩 대신 단어 임베딩을 사용하는 경우
  - REDUCE\_MEAN
    - 인코딩 Layer의 output에 평균 Pooling을 취한다.
  - REDUCE\_MAX
    - 인코딩 Layer의 output에 최대 값 Pooling을 취한다.
  - REDUCE\_MEAN\_MAX
    - 평균 pooling과 Max pooling을 개별적으로 수행한 후 두개의 벡터를 연결하여 1536 차 원의 문장 임베딩을 생성한다.
  - CLS TOKEN or FIRST\_TOKEN
    - [CLS] 토큰을 얻는다.
  - SEP\_TOKEN or LAST\_TOKEN
    - [SEP] 토큰을 얻는다.

# bert-as-service란

- fine-tuning된 BERT 모델 사용 방법
  - model 디렉토리 하위에 bert\_model.ckpt
    - 사전 훈련된 가중치가 포함된 tensorflow 체크 포인트
    - bert\_model.ckpt라는 이름의 파일이 3개 필요함
    - 위의 파일을 tensorflow로 학습시켜야 생성이 된다.
  - vocab.txt
    - WordPiece를 단어 ID에 매핑하기 위한 vocab 파일
  - bert\_config.json
    - 모델의 하이퍼파라미터를 지정하는 구성 파일)



A file explorer window showing the contents of a directory. The files listed are: bert\_config.json, bert\_model.ckpt.data-00000-of-00001, bert\_model.ckpt.index, bert\_model.ckpt.meta, and vocab.txt. The first four files are grouped together, indicating they are part of the same file system entry (the .ckpt file).

- bert\_config.json
- bert\_model.ckpt.data-00000-of-00001
- bert\_model.ckpt.index
- bert\_model.ckpt.meta
- vocab.txt

# Test 시도

- Window 환경에서 conda 가상환경을 설치하고 BERT as Service를 install하여 Test
  - Server 생성에 성공하였음
  - Client 코드를 작성하여 Embedding Vector를 얻을 수 있었다.

```
2023-03-07 19:15:24.299081: W tensorflow/core/common_runtime/gpu/gpu_device.cc:1641] Cannot dlopen some GPU libraries. Please make sure the missing libraries mentioned above are installed properly if you would like to use GPU. Follow the guide at https://www.tensorflow.org/install/gpu for how to download and setup the required libraries for your platform.
Skipping registering GPU devices...
2023-03-07 19:15:24.391030: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1159] Device interconnect StreamExecutor with strength 1 edge matrix:
2023-03-07 19:15:24.391181: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1165] 0
2023-03-07 19:15:24.391891: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1178] 0: N
I:-[33mWORKER-0-[0m:ready and listening!
I:-[35mVENTILATOR-[0m:all set, ready to serve request!
I:-[35mVENTILATOR-[0m:new config request req id: 1 client: b'84fad041-cd1a-4faf-b7f2-96249119e796'
I:-[32mSINK-[0m:send config client: b'84fad041-cd1a-4faf-b7f2-96249119e796'

>>> from tensorflow.python.client import BertClient
>>> bert_client = BertClient()
>>> bert_client.encode(['안녕하세요'])
array([[[-4.39525455e-01, -1.60576016e-01, 1.55079558e-01,
        -6.15049720e-01, 3.30122143e-01, -1.83619976e-01,
        2.14376092e-01, 6.07581496e-01, -7.56100237e-01,
        -5.40899456e-01, -4.67124917e-02, 2.78574914e-01,
        1.47827581e-01, -1.79023534e-01, -8.11148942e-01,
        -1.06246591e-01, 1.82253852e-01, 1.59852311e-01,
        -6.64056182e-01, -3.46291155e-01, -1.21130741e+00,
        1.62612230e-01, 6.33744121e-01, 3.85339648e-01,
        3.44364434e-01, -8.32074285e-01, 1.61422655e-01,
        -0.393106, 0.07640217]], dtype=float32)]
```

# Test 시도

- Window환경에서 WSL과 Docker를 이용해 Ubuntu를 설치하고, Ubuntu에서 BERT as Service를 install하여 Test
  - Server 생성에 성공하였음

```
:GRAPHOPT:[gra:opt: 53]:model config: /tmp/multi_cased_L-12_H-768_A-12/bert_config.json
:GRAPHOPT:[gra:opt: 56]:checkpoint: /tmp/multi_cased_L-12_H-768_A-12/bert_model.ckpt
:GRAPHOPT:[gra:opt: 60]:build graph...
:GRAPHOPT:[gra:opt:132]:load parameters from checkpoint...
:GRAPHOPT:[gra:opt:136]:optimize...
:GRAPHOPT:[gra:opt:144]:freeze...
:GRAPHOPT:[gra:opt:149]:write graph to a tmp file: /tmp/tmpb9z58f73
:VENTILATOR:[__i:__i: 75]:optimized graph is stored at: /tmp/tmpb9z58f73
:VENTILATOR:[__i:_ru:129]:bind all sockets
:VENTILATOR:[__i:_ru:133]:open 8 ventilator-worker sockets
:VENTILATOR:[__i:_ru:136]:start the sink
:SINK:[__i:_ru:306]:ready
:VENTILATOR:[__i:_ge:222]:get devices
Z
[]+ Stopped          bert-serving-start -model_dir /tmp/multi_cased_L-12_H-768_A-12/
test) user@DESKTOP-R0401AE:~$
```

# Environment

- BERT as Service에서 Server를 생성하기 위한 환경
  - Python==3.6.x
  - tensorflow==1.15.0
- BERT as Service에서 Client를 생성하기 위한 환경
  - Python>=3.6.x
  - tensorflow 모듈을 필요하지 않음