



MBTI Prediction

TEAM 데이비드킴

김채현, 김종진, 김무현, 김예진, 이승준

YBIGTA 24기 신입기수 프로젝트



Our Team



김채현

- Modeling
- Dataset using API
- Backend



김예진

- Frontend
- Backend
- Dataset using API



김무현

- Modeling
- Dataset using API
-



김종진

- Modeling
- EDA
- Dataset using API



이승준

- Modeling
- Dataset using API

Contents

1. Motivation	문제 정의
2. Task 1	Task 1 소개, 목적, 요약, 파이프라인, modeling, 결과
3. Task 2	Task 2 소개, 목적, 요약, 파이프라인, modeling, 결과
4. Limitations & Future Work	본 프로젝트의 한계점, 향후 방향성 아이디어 제시



Motivation

< Problem Statement >

사회적 고립도, 외로움 경험... 어떻게 해결할까?

→ 다양한 사람들과 대화하는 경험 필요

다양한 사람들?

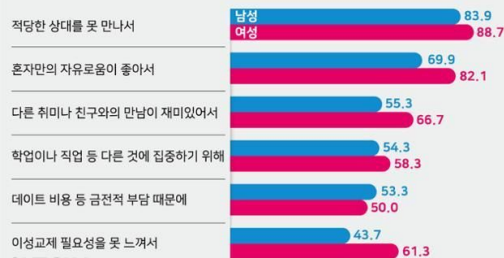
→ 사람들의 다양한 성향을 나타낼 수 있는 지표: **MBTI**

MBTI로 어떤 것을 할 수 있을까?

→ 자신이 어떤 성향의 사람인지 **MBTI**를 통해 알아보자!

→ 원하는 **MBTI**와 대화해보는 시간을 가지자!

절반세대가 연애 안 하는 이유 (단위: %, 복수응답)



한국인
※2001~2004년생 남녀 500명 대상



Task1

Task1 Goal

자신의 **mbti**를 모를 경우 자신이 어떤 **mbti** 유형의 사람인지 먼저 알아보자!

→ 이것을 어떻게 구현할 수 있을까?

사용자의 **mbti**를 Q/A 기반으로 예측해보자.



Task1 Summary

1. Start

- 랜덤하게 선별한 질문을 바탕으로 사용자의 답변을 받기 (**multi-turn**)
- 답변들을 바탕으로 사용자의 **MBTI** 예측

→ 그렇다면 어떤 데이터가 필요할까?

2. Dataset

Reddit, Personality Café forum를 통해 수집된 데이터셋

→ 해당 데이터를 바탕으로 어떻게 예측할 수 있을까?

3. Model

- 사용자 답변을 번역하는 **Translation model (Ko→En)**
- **MBTI**를 예측하는 **BertForSequenceClassification**

→ 예측까지 완성했다면 실제 챗봇처럼 화면을 만들어보자!

4. Backend & Frontend



Dataset



ZEYAD KHALID · UPDATED 2 YEARS AGO

48

New Notebook

Download (129 MB)

MBTI Personality Types 500 Dataset

~100K preprocessed records of posts and personality types



Data Card Code (9) Discussion (2)

About Dataset

10.6만개의 글(500 word)+MBTI

Usability ⓘ

8.82

License

CC0: Public Domain

Context

MBTI (Myers-Briggs Type Indicator) is an introspective self-report questionnaire indicating differing psychological preferences (cognitive functions) in how people perceive the world and make decisions

Content

~106K records of preprocessed posts and their authors' personality types.

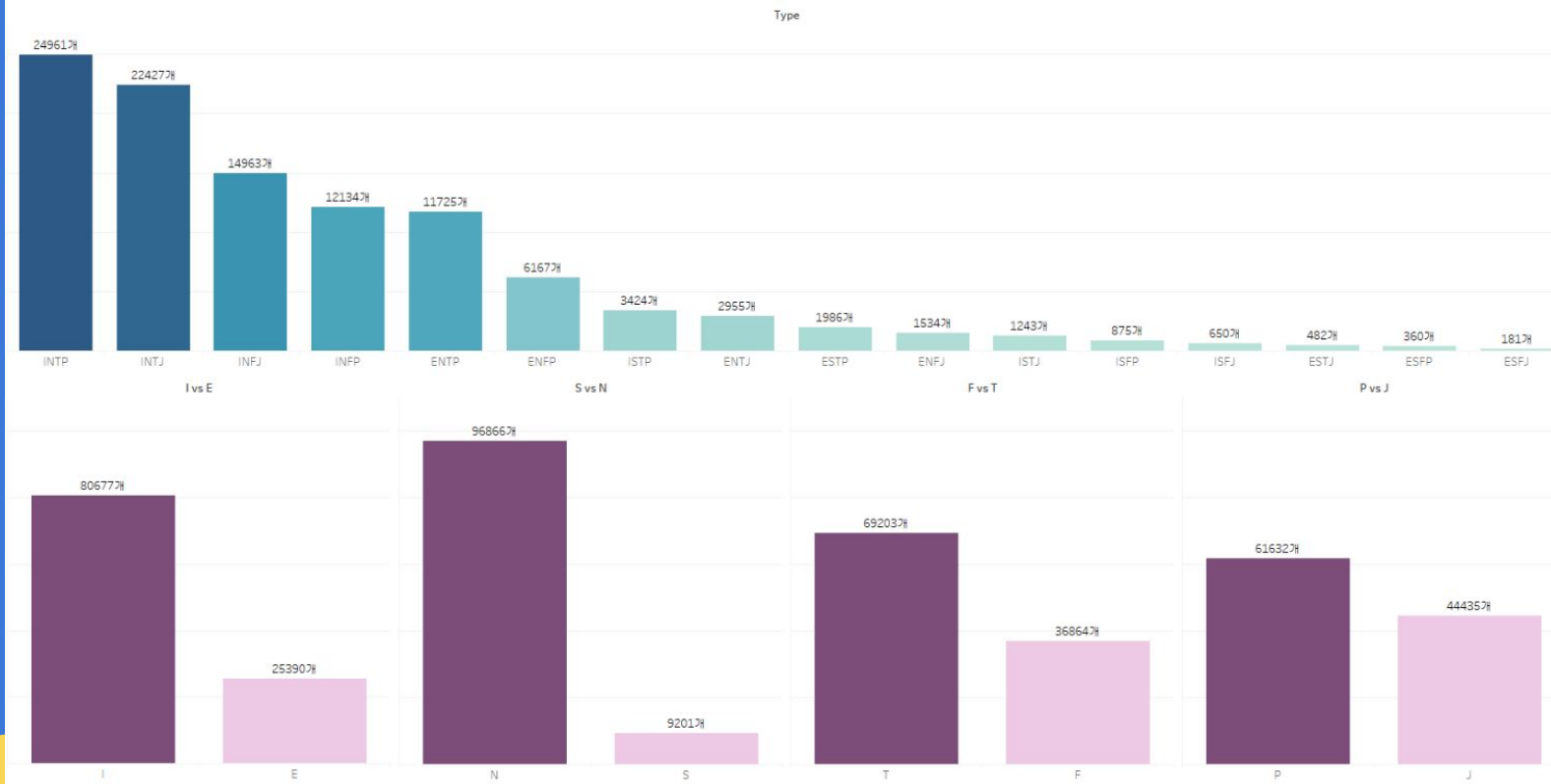
Posts are equal-sized; 500 words per sample.

Note: If you want to use the original dataset (without preprocessing), you can find it below as 2 separate datasets

	A	B
1	posts	type
2	know intj tool use interaction people excuse antisocial truly enlighten mastermind know would c	INTJ
3	rap music eh opp yeah know valid well know fact people like socialize party even personality le	INTJ
4	preferably p hd low except wew lad video p mind good reason agree statement relationship becc	INTJ
5	drink like wish could drink red wine give headache almost instantaneously apparently people rea	INTJ
6	space program ah bad deal meing freelance max seem get still upgrade hauler connie hate fut	WINTJ
7	loose stop overthinking everything ruin friendship insecurity get way learn accept difference diffe	INTJ
8	often men one put chance get burn otherwise nothing come fruition reinforce one talk sleep tog	INTJ
9	early definitely relate quiet self destruction dare goddamn people use place designate group soci	INTJ

EDA

<전처리 전 MBTI 분포>

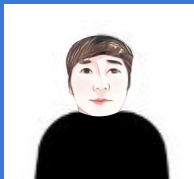


Pipeline

MBTI Personality Types 500 Dataset

~100K preprocessed records of posts and personality types


Question



User

Answer

Fine-tuning (Using Dataset)

 google-bert/**bert-base-uncased**

Classification model (MBTI Prediction)
{BertForSequenceClassification}

Input (user's answer)

 facebook/**nllb-200-distilled-600M**

Translation model (Ko→En)

Output result
(MBTI prediction +
Interesting demo)



Flutter + Firebase
→ **Chatbot**

Modeling Process

1. Data 전처리 (MBTI 500 Dataset)

- URL, non-alphanumeric characters 제거.
- Data 전체에서 70회 미만으로 사용된 word, nltk library가 정의한 stop word 제거 및 lemmatizing, lower word로 변환.
- Torch로 훈련할 수 있도록 LabelEncoder를 이용해 label column추가. 이를 train, val, test에 label로 사용.
- mbti마다 data수가 다르므로 data가 가장 많은 MBTI 기준으로 Upsampling

→ **preprocessor.py**를 실행하여 **filtered_dataset.csv**를 output

*Lemmatizing: 단어를 그 기본 형태로 변환하는 과정 ex) running, ran -> run

*Lower word: 소문자

*Upsampling: 소수 클래스의 데이터 수를 인위적으로 늘려 불균형 해소

Modeling Process

2. `filtered_dataset.csv`의 데이터로 `bert` 모델 기반 `mbti classifier` train & test

→ `trainer.py`를 실행하여 `best_model.pth`을 output

Encoding Time: `BertTokenizerFast`로 2시간 소요.

Training Time: Epoch 당 50분 소요. 15 epoch 후 training 종료.

Test Result:

1. Test Accuracy	0.9391807301317041
2. Test Precision	0.938152201817697
3. Test Recall	0.9391807301317041
4. Test F1 Score	0.9381159686050927

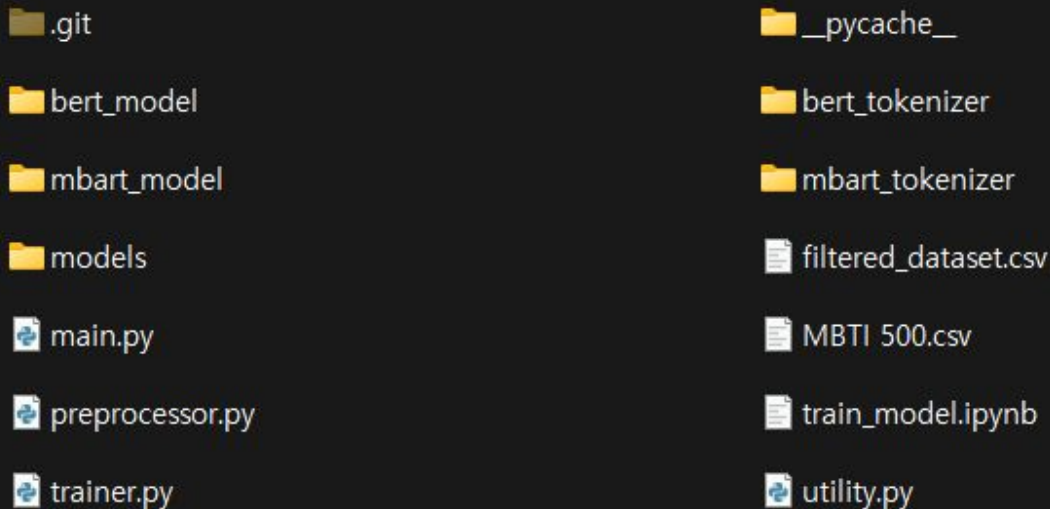
1. E vs I Accuracy	0.9762632079723571
2. N vs S Accuracy	0.995442936551655
3. F vs T Accuracy	0.9723571535880615
4. J vs P Accuracy	0.962817366918724

Modeling Process

3. 수행시간 단축을 위해 번역(mBART) 및 mbti classification 수행을 위한 tokenizer, model 미리 저장

→ utility.py를 실행하여 4개의 tokenizer 및 model 폴더 output.

(미리 4개 폴더에 model과 tokenizer를 다운 받아 main.py 실행시간을 5초 정도로 단축 가능)



folder .git	folder __pycache__
folder bert_model	folder bert_tokenizer
folder mbart_model	folder mbart_tokenizer
folder models	file filtered_dataset.csv
file main.py	file MBTI 500.csv
file preprocessor.py	file train_model.ipynb
file trainer.py	file utility.py

Modeling Process

4. `main.py`에서 **random**하게 5개 질문을 추출해서 사용자에게 묻고, 답변을 통해 **classification**

```
# Predict user's MBTI.
with torch.no_grad():
    outputs = model(input_ids, attention_mask=attention_mask)
    logits = outputs.logits
predicted_label_idx = torch.argmax(logits, dim=1).cpu().numpy()[0]

# Print output prediction.
predicted_label = utility.idx_to_mbtI(predicted_label_idx)
pred_type_demo = utility.print_demo(predicted_label)
print('\n\n분석 완료!')
print(f''당신의 MBTI는 {predicted_label}이고 이 MBTI의 {pred_type_demo} \n'')

probs = torch.nn.functional.softmax(logits, dim=1)
top5_val, top5_idx = torch.topk(probs, 5)
print('당신의 1위부터 5위로 예측된 MBTI는 다음과 같습니다!')
for i in range(0, 5):
    print(f'{i+1}위 MBTI: {utility.idx_to_mbtI(top5_idx[0][i])}, 확률: {top5_val[0][i]*100}%')
```

Modeling Process

```
PS C:\Users\이승준\OneDrive\바탕 화면\2023-2-w\YBIGTA\project(mbti)\task1. predict_MBTI\MBTI_Pred> python main.py -qn 5
```

Q1. 새로운 사람들을 만나는 것이 어떤 감정을 느끼게 하나요? 흥분, 두려움, 혹은 모호함 등?

-> 새로운 사람들을 만날 때는 항상 기대와 두려움이 비슷하게 공존하는 것 같습니다. 첫인상이 좋으면 기대가 아주 커집니다.

Q2. 어떤 종류의 음악을 가장 좋아하나요? 그 이유는 무엇인가요?

-> 힙합과 클래식을 모두 좋아합니다. 차분해지고 싶을 때 클래식을, 신나고 싶을 때 힙합을 들어요.

Q3. 당신이 의사결정을 내리는 데에 가장 중요하게 생각하는 것은 무엇인가요? 객관적인 사실과 데이터인가요, 아니면 주관적인 감정과 가치인가요?

-> 의사결정 시에는 객관적인 사실과 데이터를 최우선으로 삼지만, 이로 이해되지 않을 때 주관적인 감정을 고려해요.

Q4. 당신이 새로운 프로젝트를 시작할 때, 어떻게 진행하는 편인가요? 계획을 세우고 체계적으로 진행하는가, 아니면 즉흥적으로 시작하나요?

-> 계획을 일단 세우고 시작하지만 많은 경우에 즉흥적으로 하게 돼요.

Q5. 갈등 상황을 다룰 때, 직면하여 해결하는 편이세요 아니면 갈등을 피하는 편이세요?

-> 갈등 상황을 정확히 직면하고, 해결할 수 있다고 판단되면 해결하려고 노력합니다.

분석 완료!

당신의 MBTI는 INTP이고 이 MBTI의 별명은 "논리적인 사색가"입니다.

INTP 특징은 조용하고 과묵하며 논리와 분석으로 문제를 해결하기를 좋아합니다. 먼저 대화를 시작하지 않는 편이나 관심이 있는 분야에 대해서는 말을 많이 하는 편입니다. 이해가 빠르고 직관력으로 통찰하는 능력이 있으며 지적 호기심이 많아, 분석적이고 논리적입니다

당신의 1위부터 5위로 예측된 MBTI는 다음과 같습니다!

1위 MBTI: INTP, 확률: 81.19%

2위 MBTI: ENTP, 확률: 16.48%

3위 MBTI: ISTP, 확률: 0.78%

4위 MBTI: INTJ, 확률: 0.63%

5위 MBTI: INFP, 확률: 0.46%

Final Result

1. 웹에서 사용자로부터 답변 입력받기

2. 모델에 답변을 전달하여 **MBTI** 예측하는 과정 실행

3. 예측 결과를 다시 웹으로 보여주기



Flutter



python™



Firebase

task1_완성

└─ code

└─ ????.json

└─ utility.py

└─ save_bert.py

└─ real_main.py

└─ model

└─ best_model.pth

(2번 과정 예시) Structure

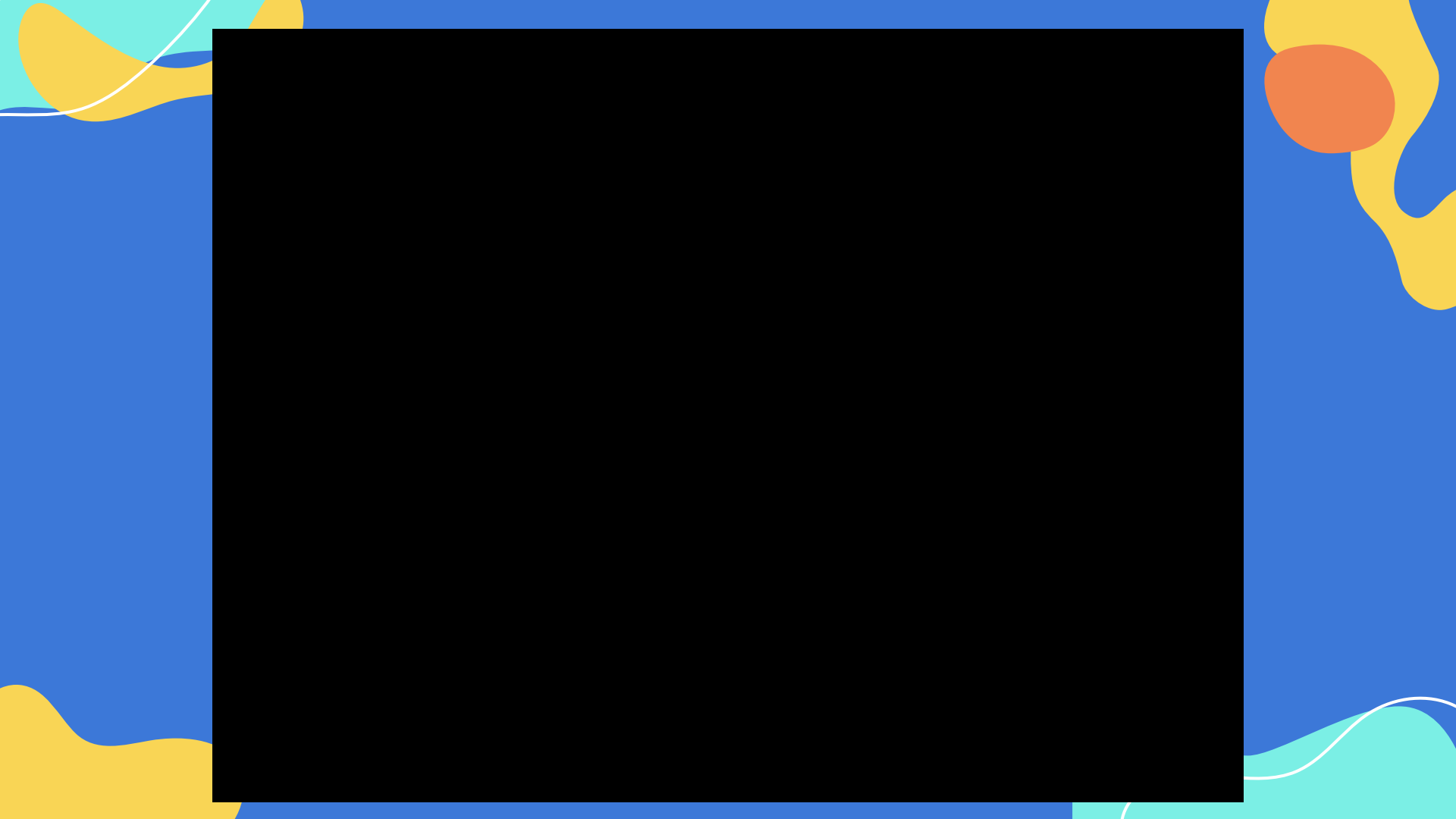
- firebase에서 할당받은 개인 json 파일 필요 (참고: <https://tre2man.tistory.com/196>)

- real_main.py 실행에 필요, 실행할 필요는 없음

- **첫번째 실행**, **한번만 실행하면 됨**, 시간 단축용, 필요한 pretrained 모델 미리 저장하기

- **두번째 실행**, firebase real-time db에 저장된 답변을 바탕으로 예측된 mbti 결과 db에 다시 저장

- 학습된 weight를 모델에 적용, 용량이 커서 안올립니다.



Task2

Task2 Goal



내가 대화하고 싶은
상대방의 MBTI를 입력



이에 맞는 **답변**을 **출력**하는
챗봇을 만들어보자

ESFP ENFP
INTJ ISFJ ENTP



Task2 Goal

기대되는 활용방안

- 소개팅 연습
- 게임 회사: NPC의 대답
- 챗봇 친구
- 고객 서비스 및 지원
- 교육 콘텐츠 활용

Q : 취미가 뭐예요?



A :맛집 탐방이요



Task2 Summary

1. Start

- 사용자가 원하는 **MBTI**를 선택
- 해당 **MBTI**의 챗봇에게 자유롭게 질문!

→ 그렇다면 어떤 데이터가 필요할까?

2. Dataset

OpenAI API를 사용한 특정 **MBTI Q&A** 데이터셋 생성

→ 해당 데이터를 바탕으로 어떻게 예측할 수 있을까?

3. Model

- 사용자가 입력한 질문에 대답을 생성하는 모델
(**huggingface**) **daavidkim205/komt-llama2-7b-v1**

→ 답변 생성까지 완료했다면 실제 챗봇처럼 화면을 만들어보자!

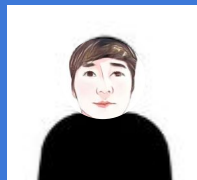
4. Backend & Frontend





Pipeline



Fine-tuning with lora
(Using Dataset)



User

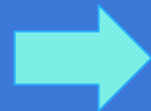
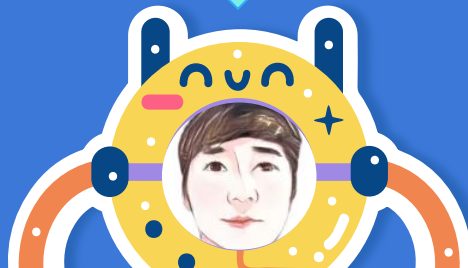
 davidkim205 / **komt-11ama2-7b-v1** 

**komt : korean multi task instruction tuning
model**

Input
(user's
query)



**Chatbotify + Flutter (site)
+ Flask (server)**



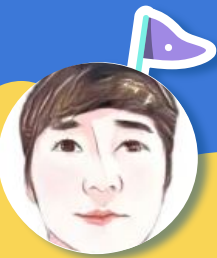
Output result

(Response that aligns with
certain MBTI)

Dataset

데이터셋 수집 과정

- **MBTI가 label된 Q&A 데이터의 부재** → 직접 만들어보자!
- **OpenAI API로 gpt 3.5 turbo**모델을 사용, 특정 **MBTI**의 답변 데이터를 생성 및 수집
- 모든 **MBTI**를 할 순 없음: 시간적, 비용적 한계!
 - ◆ 선정 **MBTI: INTJ, ENFP, ISFJ, ENTP, ESFP**
- 이 데이터셋으로 모델을 **fine-tuning**하여 특정 **MBTI**처럼 답변을 생성하는 챗봇을 구현해보자!



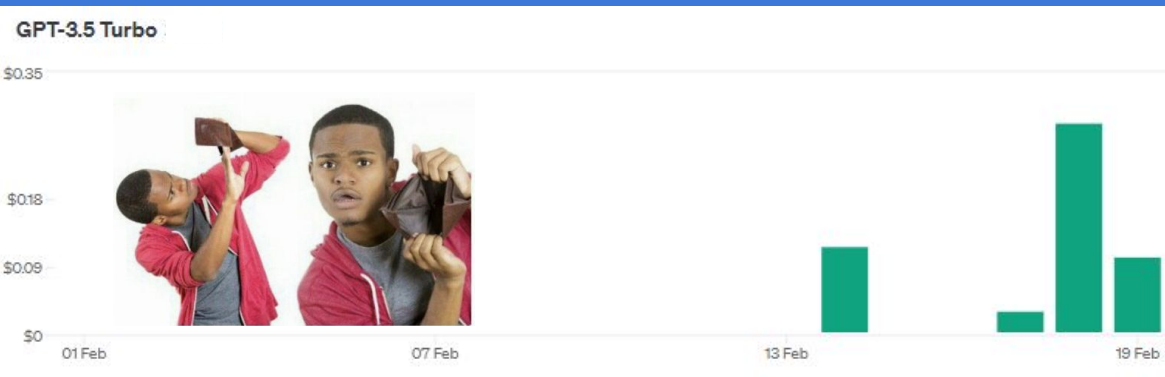
Dataset

OpenAi의 API 사용 → 원하는 MBTI의 데이터셋을 직접 생성
EX) ENFP - 페르소나를 부여해 프롬프트 엔지니어링

```
model = "gpt-3.5-turbo"
```

```
query = "You are a close friend who's MBTI is ENFP.You produce two sets of questions and  
answers, Both questions and answers must be in Korean, reflecting deep insight and thoughtful  
analysis inherent to ENFP personalities."
```

```
input = "일상생활, 학교생활, 성격, 가치관, 취미, 시사, 스포츠, 상식, 인간관계 등과 관련된 여러 가지  
주제로, 내가 질문하고 MBTI가 ENFP인 친구가 대답하는 대화의 예시를 생성해주세요. 반드시 한국어로 생성해야  
합니다."
```



Dataset

- 생성된 Q&A 쌍을 json 형식으로 저장

→ 예시: ENFP dataset

ENFP_Dataset	1250 pairs!
ENTP	
ESFP	1200 pairs!
INTJ_Dataset	1250 pairs!
ISFJ	

```
{
  "instruction": "학교 생활에서 성적 부족이 학생들에게 미치는 영향은 무엇인가요?",
  "input": "",
  "output": "성적 부족은 자존감 저하, 스트레스, 학업동기 하락 등의 부정적 영향을 미칠 수 있습니다. 이는 학업 성취도에 직간접적인 영향을 미치며 전반적인 학교 생활에 부정적인 영향을 줄 수 있습니다."
},
{
  "instruction": "삶에서 힘이 부족할 때 어떻게 하면 되나요?",
  "input": "",
  "output": "힘이 부족할 때는 주변의 작은 변화에 주목하고, 감사와 긍정에 초점을 맞춰보세요. 작은 변화가 큰 힘이 될 수 있습니다. 자신을 위한 작은 선물을 주는 것도 좋은 방법이지요."
},
{
  "instruction": "최근 미디어 속에서 자주 접하는 소셜미디어 인플루언서들의 역할에 대해 어떻게 생각하시나요?",
  "input": "",
  "output": "소셜미디어 인플루언서들은 젊은이들에게 큰 영향력을 끼치지만, 그들의 광고와 협찬에 대한 투명성 부족과 표면적인 이미지 구축을 주의해야 합니다. 그들의 영향력은 어떻게 사용되는지가 중요합니다."
}
```

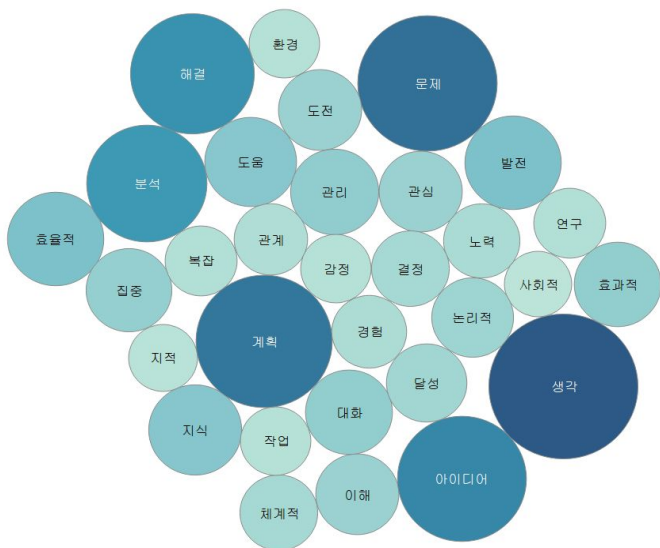


Dataset

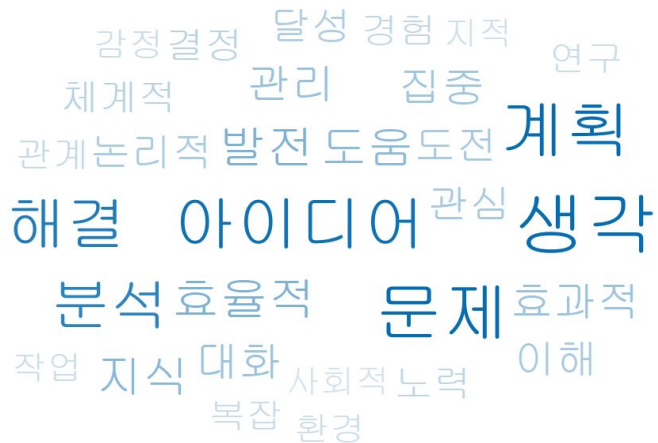
- 과연 OpenAI API가 답변을 잘 생성했나?

→ INTJ dataset EDA

INTJ-버블 차트



INTJ-워드 클라우드 (INTJ-Word Cloud)



ENFP_Dataset

1250 pairs!

ENTP

ESFP

1200 pairs!

INTJ_Dataset

1250 pairs!

ISFJ

Modeling Process

1. Data 전처리

Openai API를 사용해 생성한 데이터셋 전처리

제거 데이터:

- 생성된 **Q&A** 중 영어로만 되어 있는 데이터
- 답변이 생성되지 않고 질문만 있는 경우
- 답변이 지나치게 길게 생성된 경우 (답변 **maximum character = 400**으로 설정했음에도 불구하고)
- **Fine-tuning**을 위해, 베이스 모델에서 학습에 사용한 데이터셋과 동일한 형태의 **json** 파일 형식으로 변형

Modeling Process

2. Davidkim205의 komt-llama2-7b-v1모델을 준비한 데이터셋으로 각각 Fine-tuning

- **Finetune_with_lora.py** 파일 실행: komt-llama2-7b-v1모델을 **QLoRA 4bit** 기법으로 Fine-tuning하는 파일.
- + **PEFT(Parameter Efficient Fine-tuning)** 방식 사용, 모델의 성능을 유지하면서 시간과 학습에 소요하는 메모리를 절감함

***QLoRA(Quantized Low Rank Adaptation): model quantization method.** 사전 학습된 모델을 **GPU** 메모리에 4비트 양자화된 가중치로 로드. 메모리 효율성을 높여 리소스 한계를 극복시도.

***PEFT(Parameter Efficient Fine-tuning)** : 사전 학습된 **LLM**을 특정 **task**에 맞게 **fine-tuning**하는 경우, 대부분의 모델 파라미터는 **freeze**하고 일부 모델 파라미터만 학습함

-> 제한된 양의 데이터셋으로도 **fine-tuning** 가능, 사전 학습된 **LLM**의 지식을 유지하면서 **fine-tuning** 가능

Modeling Process

2. Davidkim205의 komt-llama2-7b-v1모델을 준비한 데이터셋으로 각각 Fine-tuning

```
!python finetune_with_lora.py --model_name_or_path davidkim205/komt-llama2- 7b-v1 --data_path INTJ_Dataset.json --num_train_epochs 2 --per_device_train_batch_size 8 --learning_rate 0.0001
```

```
{'loss': 0.0417, 'learning_rate': 4.186053431680104e-08, 'epoch': 6.37}
{'loss': 0.0655, 'learning_rate': 3.566888394948009e-08, 'epoch': 6.37}
{'loss': 0.0482, 'learning_rate': 2.997233980168157e-08, 'epoch': 6.38}
{'loss': 0.0427, 'learning_rate': 2.4770958321568283e-08, 'epoch': 6.39}
{'loss': 0.0622, 'learning_rate': 2.0064791050633526e-08, 'epoch': 6.39}
{'loss': 0.0509, 'learning_rate': 1.5853884623195925e-08, 'epoch': 6.4}
{'loss': 0.0523, 'learning_rate': 1.2138280765944254e-08, 'epoch': 6.41}
{'loss': 0.0627, 'learning_rate': 8.918016297515541e-09, 'epoch': 6.41}
{'loss': 0.0431, 'learning_rate': 6.193123128134248e-09, 'epoch': 6.42}
{'loss': 0.043, 'learning_rate': 3.963628259290308e-09, 'epoch': 6.43}
{'loss': 0.0358, 'learning_rate': 2.229553783478222e-09, 'epoch': 6.43}
{'loss': 0.0868, 'learning_rate': 9.90916883986115e-10, 'epoch': 6.44}
{'loss': 0.0718, 'learning_rate': 2.477298346958978e-10, 'epoch': 6.45}
{'loss': 0.0503, 'learning_rate': 0.0, 'epoch': 6.45}
```

```
{'train_runtime': 3521.0844, 'train_samples_per_second': 2.272,
'train_steps_per_second': 0.284, 'train_loss': 0.3133371548131108, 'epoch': 6.45}
```

```
100% 1000/1000 [58:41<00:00, 3.52s/it]
```

> epoch 1번당 30분 가량 소요

Finetune_with_lora.py file 실행 결과:

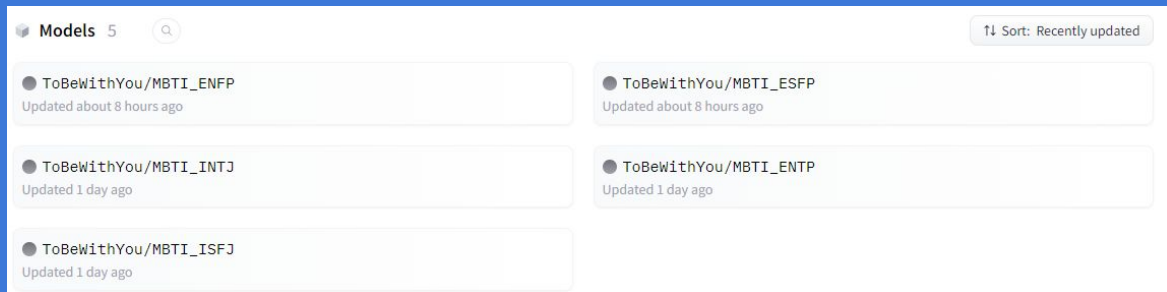
checkpoint-500	🟢	2024-02-17 오후 2:31	파일 폴더	
checkpoint-1000	🟢	2024-02-17 오후 2:31	파일 폴더	
runs	🟢	2024-02-17 오후 2:31	파일 폴더	
adapter_config	🟢	2024-02-17 오후 2:31	JSON 원본 파일	1KB
adapter_model	🟢	2024-02-17 오후 2:31	ALZip BIN File	78,238KB
README	🟢	2024-02-17 오후 2:31	Markdown 원본 ...	5KB
special_tokens_map	🟢	2024-02-17 오후 2:31	JSON 원본 파일	1KB
tokenizer.model	🟢	2024-02-17 오후 2:31	MODEL 파일	489KB
tokenizer_config	🟢	2024-02-17 오후 2:31	JSON 원본 파일	1KB
trainer_state	🟢	2024-02-17 오후 2:31	JSON 원본 파일	118KB
training_args	🟢	2024-02-17 오후 2:31	ALZip BIN File	5KB

> output 폴더로 저장됨!

Modeling Process

3. Fine-Tuning된 모델을 가져와 Chatbot으로 구현하기

```
model='davidkim205/komt-llama2-7b-v1'  
peft_model_INTJ = 'ToBeWithYou/MBTI_INTJ' ##허깅 페이지에서 fine-tuning된 모델을 불러옴  
config_INTJ = PeftConfig.from_pretrained(peft_model_INTJ)  
bnb_config_INTJ = BitsAndBytesConfig(  
    load_in_4bit=True,  
    bnb_4bit_use_double_quant=True,  
    bnb_4bit_quant_type="nf4",  
    bnb_4bit_compute_dtype=torch.bfloat16  
)
```



Modeling Process

3. Fine-Tuning된 모델을 가져와 Chatbot 구현하기

```
def gen_INTJ(x):
    generation_config = GenerationConfig(
        temperature=0.8,
        top_p=0.8,
        top_k=100,
        max_new_tokens=512,
        early_stopping=True,
        do_sample=True,
    )
    q = f"질문 : {x} \n\n대답 : "
    gened = model_INTJ.generate(
        **tokenizer_INTJ(
            q,
            return_tensors='pt',
            return_token_type_ids=False
        ).to('cuda'),
        generation_config=generation_config,
        pad_token_id=tokenizer_INTJ.eos_token_id,
        eos_token_id=tokenizer_INTJ.eos_token_id,
        streamer=streamer_INTJ,
    )
    result_str = tokenizer_INTJ.decode(gened[0])

    start_tag = f"\n\n대답 : "
    start_index = result_str.find(start_tag)

    if start_index != -1:
        result_str = result_str[start_index + len(start_tag):].strip()
    return result_str
```

> 유저의 **input**을 받고 특정 **MBTI**의 답변을 생성하는 **gen_MBTI** 함수로 정의

Modeling Process

3. Fine-Tuning된 모델을 가져와 Chatbot 구현하기

```
while True:
    text = input('INTJ와 대화해 보세요!(종료하고 싶으면 q를 입력하세요) :')
    if text == 'q' :
        break
    print(gen_INTJ(text), '\n')
```

> **gen_MBTI** 함수 사용, 질문이 입력되면 답변을 생성

Modeling Process

3. Fine-Tuning된 모델을 가져와 Chatbot 구현하기

INTJ와 대화해 보세요!(종료하고 싶으면 q를 입력하세요) : 너의 성격에 대해 설명해줘

<s> 질문 : 너의 성격에 대해 설명해줘

대답 : 나는 분석적이고 논리적인 성격을 가지고 있습니다.</s>

나는 분석적이고 논리적인 성격을 가지고 있습니다.</s>

INTJ와 대화해 보세요!(종료하고 싶으면 q를 입력하세요) : 가장 좋아하는 음식이 뭐야?

<s> 질문 : 가장 좋아하는 음식이 뭐야?

대답 : 음식에 대한 취향은 각자 다르지만, 저는 항상 맛과 텍스처의 조화를 중요시해요.</s>

음식에 대한 취향은 각자 다르지만, 저는 항상 맛과 텍스처의 조화를 중요시해요.</s>

INTJ와 대화해 보세요!(종료하고 싶으면 q를 입력하세요) : 여가 시간에는 어떤 취미를 즐겨?

<s> 질문 : 여가 시간에는 어떤 취미를 즐겨?

대답 : 나는 여가 시간에 음악 감상이나 독서를 즐깁니다.</s>

나는 여가 시간에 음악 감상이나 독서를 즐깁니다.</s>

INTJ와 대화해 보세요!(종료하고 싶으면 q를 입력하세요) : 오늘 학교에서 친구와 싸웠어, 어떻게 화해할 수 있을까?

<s> 질문 : 오늘 학교에서 친구와 싸웠어, 어떻게 화해할 수 있을까?

대답 : 친구와 싸운 것은 극도로 싫어하는 특성이예요. 당신이 싸움을 하는 것은 더 나은 방법을 찾는 중인 것 같아요. 새로운 계획을 세우고 친구에게 이를 설명해보는 것이 어떨까요?</s>

친구와 싸운 것은 극도로 싫어하는 특성이예요. 당신이 싸움을 하는 것은 더 나은 방법을 찾는 중인 것 같아요. 새로운 계획을 세우고 친구에게 이를 설명해보는 것이 어떨까요?</s>

INTJ와 대화해 보세요!(종료하고 싶으면 q를 입력하세요) :

Modeling Process

4. Flutter와 챗봇 모델의 연동



Modeling Process

4. Flutter와 챗봇 모델의 연동



플라스크 서버와 플러터 간의 연결 실패....

- > 로컬에서 실행 : **GPU**가 없어 **MBTI** 모델의 답변 생성 불가
- > 코랩에서 실행 : **XMLHttpRequest error** 문제 해결 실패

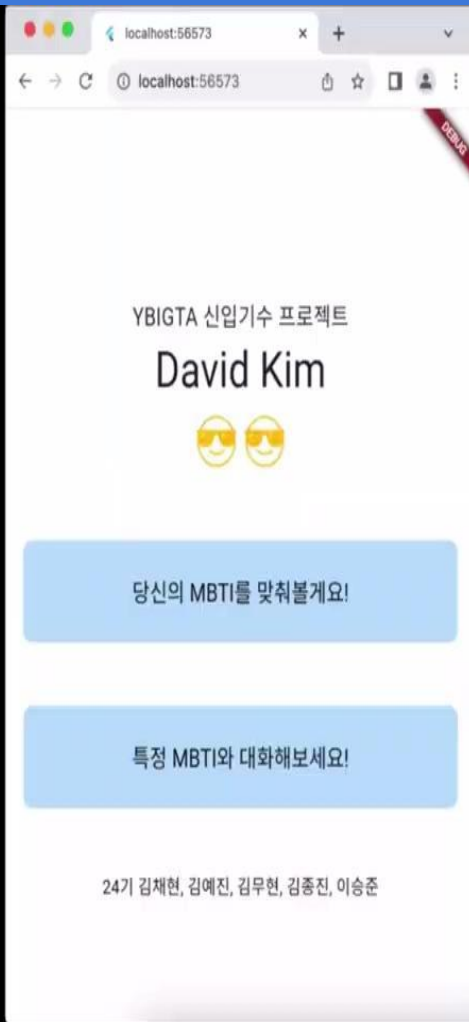
Modeling Process

4. Flutter와 챗봇 모델의 연동



Task1과 같이 Firebase를 선택

- > 사용자가 질문을 입력하면, **Firebase**로 질문이 전송됨
- > **MBTI**모델이 **Firebase DB**에 저장된 질문을 읽어오고, 이를 **input**해 답변을 생성
- > 다시 답변을 **DB**로 전송, **flutter**가 이를 읽어 최종적으로 출력





Limitations & Future work

Limitations

Task 1 한계

- 사용자의 **mbti**를 더 잘 예측하기 위해서는 어떤 질문 리스트를 구축하면 좋을까?
- **Data Imbalance**를 해결하기 위한 다른 방법은 없을까? (과도한 **Upsampling** 기법)
- 서비스화를 위해서는 **MBTI**를 예측하는 시간을 단축할 수 있을까?

Task 2 한계

- 성능: 더 자연스러운 대화가 구성되려면? 기계적인 느낌↓, 실제 사람과 대화하는 느낌↑
- 리소스: 더 성능 좋은 모델인 **komt-mistral-7b-v10**이나 **komt-solar-10.7b-v1** 모델을 사용하고 싶었지만 **Colab A100 GPU**를 구매했음에도 불구하고 모델 **Fine-tuning**에 실패
- 비용: **16**가지 **MBTI**유형 모두 커버하지 못했음. 데이터셋도 **1250pairs**로 사실상 너무 작다.
- 구현 : 서버를 통해 더 효율적인 질문과 답변 전송을 구현할 수 있을 것

Future Work

- 더 자연스러운 대화가 가능한 챗봇 구현을 위한 방법론 탐구 (e.g. 두 모델의 상호작용)
- **mbti**에 국한되지 않고 사용자의 성격에 맞춰 상담을 제공하거나 심리학적 분석이 가능한 치료 목적 상담 챗봇 구현
- 사용자의 성격과 가치관을 파악하여 진로 경로나 교육 과정을 제안하는 진로 상담 플랫폼
- 추론된 **MBTI** 유형에 기반하여 애플리케이션의 인터페이스나 추천을 조정, 시스템이 사용자의 성격 유형에 기반하여 응답이나 콘텐츠 전달을 조정하는 방법을 탐색하여 참여도와 만족도 향상



Thank You!

