「나의 AI찐친」

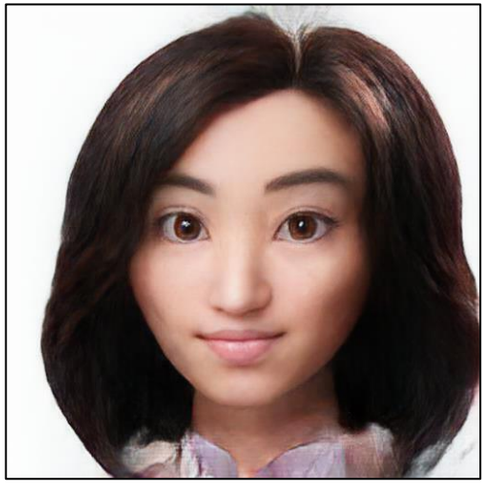
자연어 처리 모델을 활용한 개인화 AI휴먼

Team 연어유희

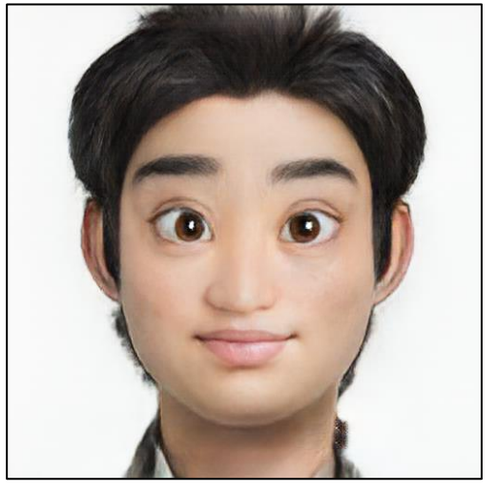
Content

팀원 소개

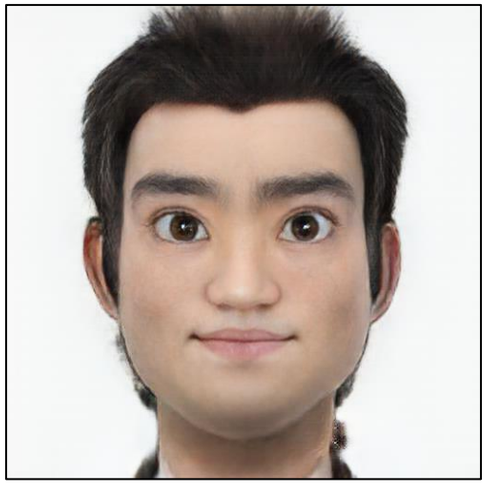
팀장
남궁지희



이소담



이 한



장세종

나보다 나를 더 잘 아는

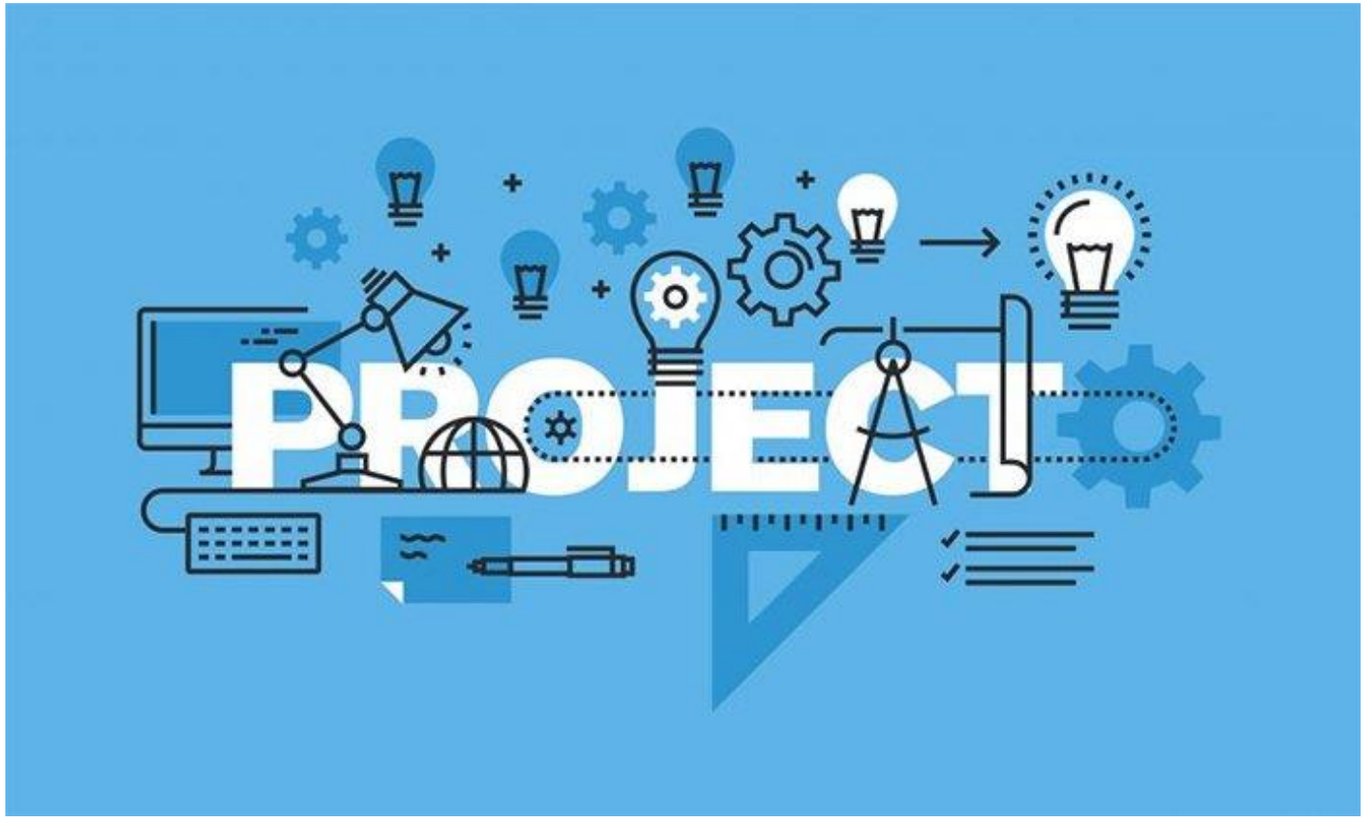
YH. Ji Hee

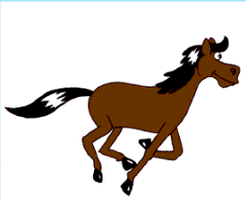
당신을 이해하고
당신에게 귀 기울이는 존재

“안녕👋

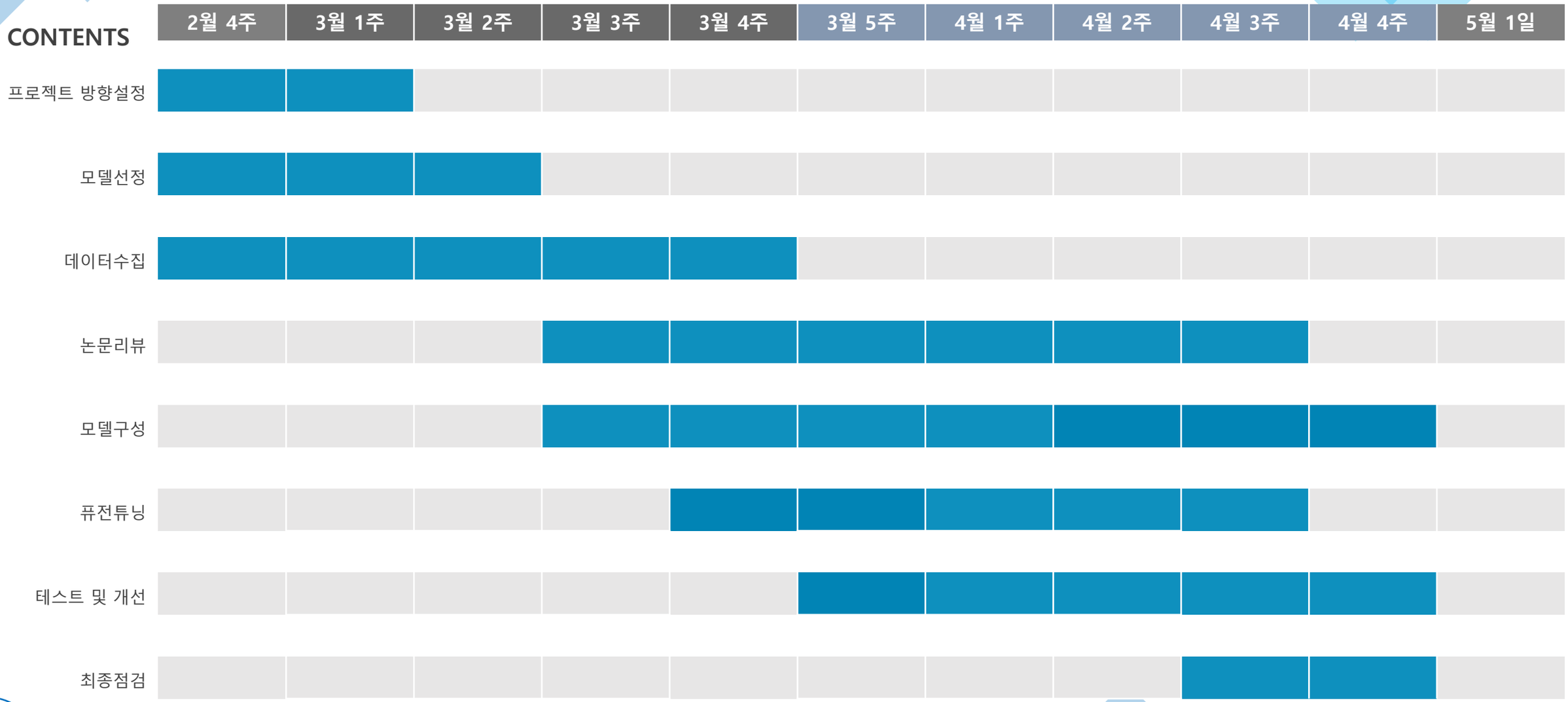
난 너의 AI찐친
유희지희야”

프로젝트 진행 상황





CONTENTS


2022. 3
2022. 4
2월 4주
3월 1주
3월 2주
3월 3주
3월 4주
3월 5주
4월 1주
4월 2주
4월 3주
4월 4주
5월 1일
프로젝트 방향설정
모델선정
데이터수집
논문리뷰
모델구성
퓨전튜닝
테스트 및 개선
최종점검

데이터 수집

..	
웰니스 대화 스크립트 데이터셋	Add files via upload
한국어 감정 정보가 포함된 연속적 대화 데이터셋	Add files via upload
한국어 대화 데이터셋	Add files via upload
ChatBotData_지희.csv	03/20
dataset1.xlsx	3/22일까지 합친 내용
kakaotalk.csv	800
트위터_대화시나리오DB_2000Set.xlsx	Add files via upload
한국어_연속적_대화_데이터셋.xlsx	Add files via upload

```
In [10]: data.iloc[:50]
```

Out[10]:

	dialog #	발화	감정
0	S	아 진짜! 사무실에서 피지 말라니깐! 간접흡연이 얼마나 안좋은데!	분노
1	NaN	그럼 직접흡연하는 난 얼마나 안좋겠니? 안그래? 보면 꼭 지 생각만 하고.	혐오
2	NaN	친구도 담배 피지?	중립
3	NaN	친구? 누구?	중립
4	NaN	몰라! 니 친구래.	중립
5	NaN	내 친구? 친구 누구?	중립
6	NaN	그걸 내가 어떻게 알아!	분노
7	S	그래서 무슨 일 해?	중립
8	NaN	그냥 암꺼나 조금.	중립
9	NaN	암꺼나? 암꺼나 뭐?	중립
10	NaN	회사에서 코딩도 하고, 있어 그런 거.	중립

```
In [13]: df=pd.DataFrame(columns=['Q','A'])

In [14]: df
```

Out[14]:

Q	A

```
In [17]: df.loc[0]=[['아 진짜! 사무실에서 피지 말라니깐! 간접흡연이 얼마나 안좋은데!',
'그럼 직접흡연하는 난 얼마나 안좋겠니? 안그래? 보면 꼭 지 생각만 하고.',
'친구도 담배 피지?'],
'친구? 누구?']

In [18]: df
```

Out[18]:

	Q	A
0	[아 진짜! 사무실에서 피지 말라니깐! 간접흡연이 얼마나 안좋은데!, 그럼 직접흡연...	친구? 누구?

데이터 가공 (감정분류)

7개 감정분류 원본 데이터

행복	중립	슬픔	공포	혐오	분노	놀람
170						

29	S	안 내켜.	혐오
30	NaN	왜 ?	중립
31	NaN	뭔가 말리는 기분이야. 불길해.	공포
32	NaN	뭐가?	중립
33	NaN	생각해봐. 친구랍시고 갑자기 불쑥 나타나서는 안이상해?	공포
34	NaN	뭐가 이상해? 우정만 돋는구만. 뭐.	중립
35	NaN	우정 좋아하네. 싸가지 못봤어? 지가 뭐라고	혐오
36	NaN	절친이라며.	중립
37	NaN	나이에 갑자기 연락하는 것도 이상하고	혐오
38	NaN	복 받은 넘인가 보지. 뭐.	중립
39	NaN	너 오늘 유난히 대답 잘 한다? 라임 맞춰가면서.	혐오
40	NaN	그니까. 쓸데없는 걱정 말고 도전해봐.	중립
41	S	집 짓는 이유가 그렇게 중 해? 왜, 내가 사기라도 칠까봐?	혐오
42	NaN	또. 험악하게. 그런 게 아니라 왜 집을 짓는지 알아야 어떤 집이 필 한지를 알지...	중립
43	NaN	아하~ 그러니까 날 잘 알고 싶다 이거지? 내가 궁금하시다? 뭐가 그렇게 궁금할까?...	중립

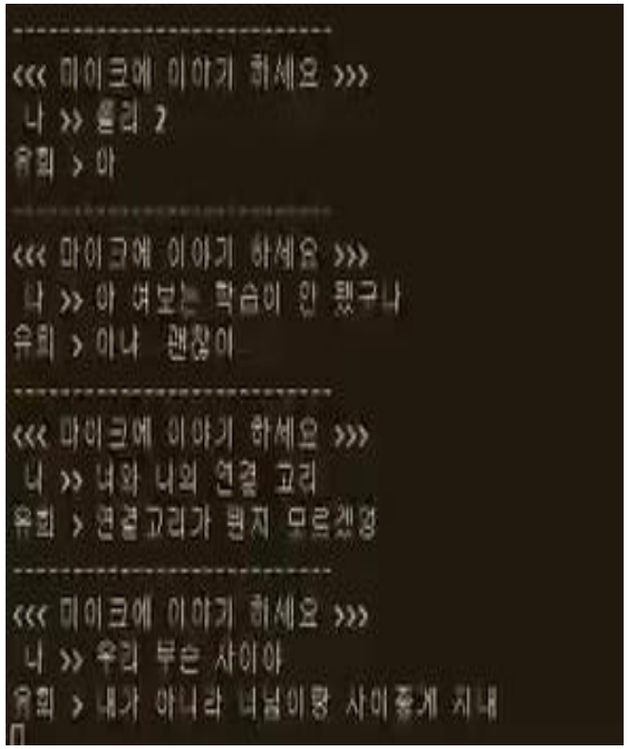
긍정, 부정, 중립 라벨링 작업

	id	document	label
404	4221289	NaN	0
412	9509970	NaN	1
470	10147571	NaN	1
584	7117896	NaN	0
593	6478189	NaN	0

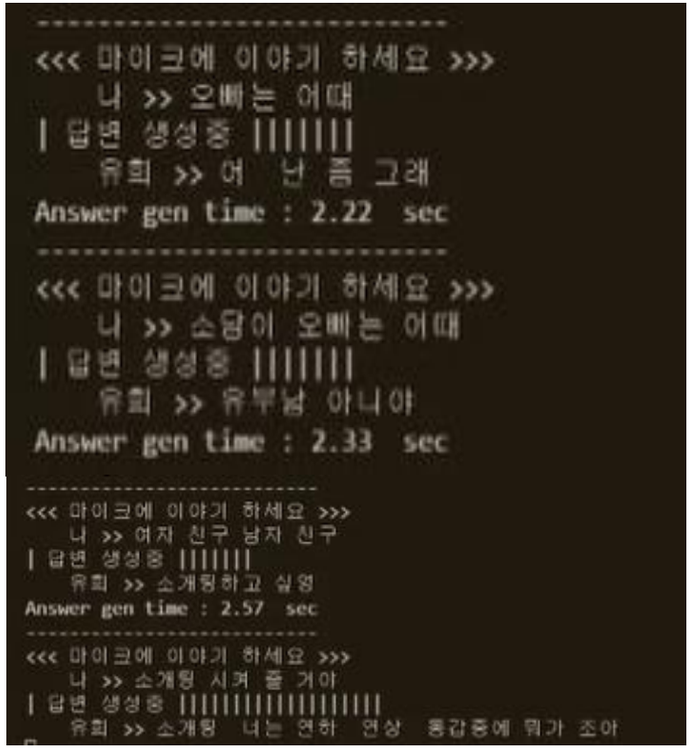
화자(**Q**) / 청자(**A**) 대화쌍 구성

```
import pandas as pd
data=pd.read_csv('data.csv',header=None)
df=pd.DataFrame(columns=['Q','A'])

__speaker='천구'
this_text=''
box=[]

for a in data[0]:

    if '[이 T🍒]' in a:
        this_speaker='천구'
    else:
        this_speaker='지희'
    w=a[::-1].find(']')
    text=a[-w+1:]

#      print(text)
    if this_speaker==__speaker:
        this_text+=' '+text

    else:
        box.append([__speaker,this_text])
        if __speaker=='천구':
            __speaker='지희'
        else:
            __speaker='천구'

        this_text=text

box.append([this_speaker,this_text])

q=[]
a=[]
for idx,i in enumerate(box):
    if i[0]=='천구':
        q.append(i[1])
    else:
        a.append(i[1])

df['Q']=q
df['A']=a

df
```

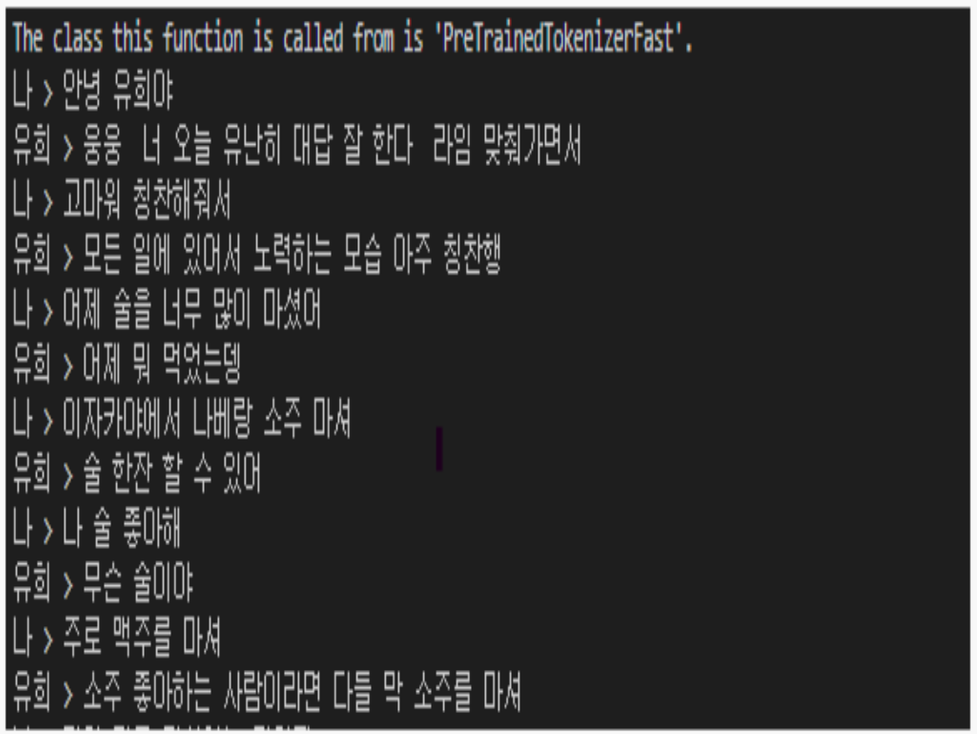
시연 결과



Epoch = 5

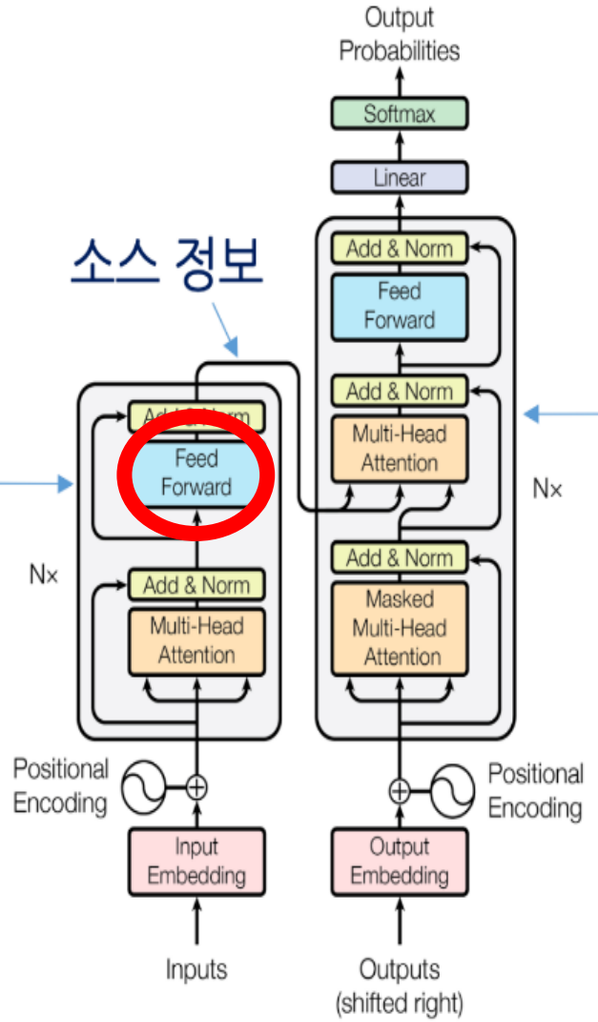


Epoch = 10



Epoch = 20

트랜스포머



소스 정보

Encoder
소스 시퀀스 압축

Decoder
타겟 시퀀스 생성

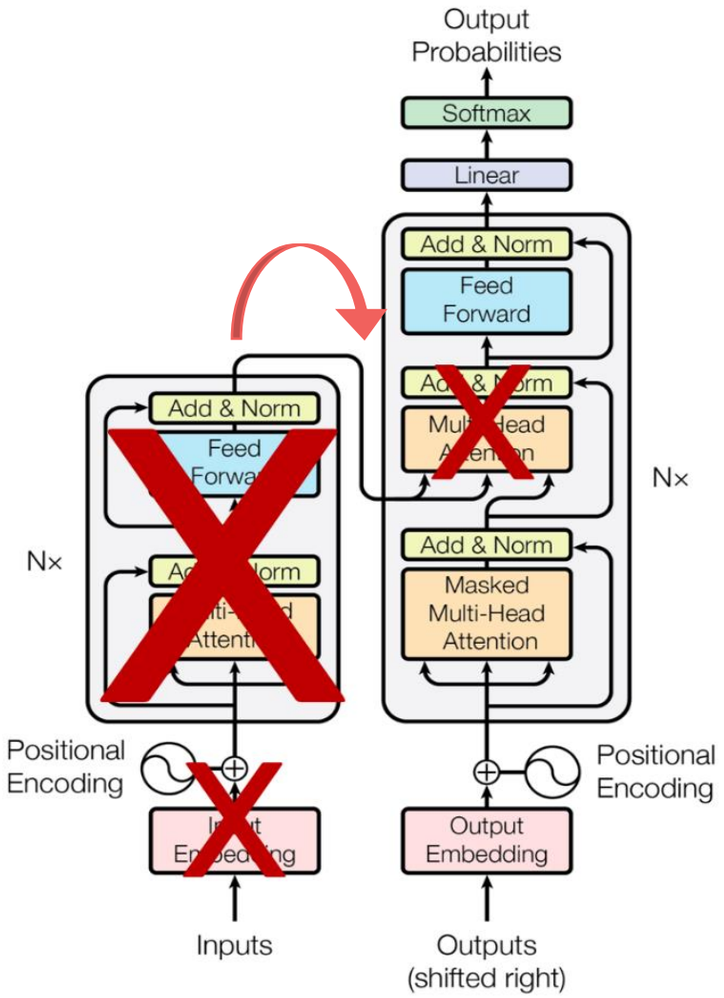
Feed Forward
- 2개의 Dense Layer와 Relu 함수로 구성

Add & Norm
- 레이어 정규화, 잔차연결 역할

Positional Encoding
문장의 의미 파악을 위해
문장의 단어 위치정보가 중요

KoGPT2 (Korean Generative Pre-trained Transformer)

GPT-2 모델을 Fine-Tuning한 한국어 언어모델

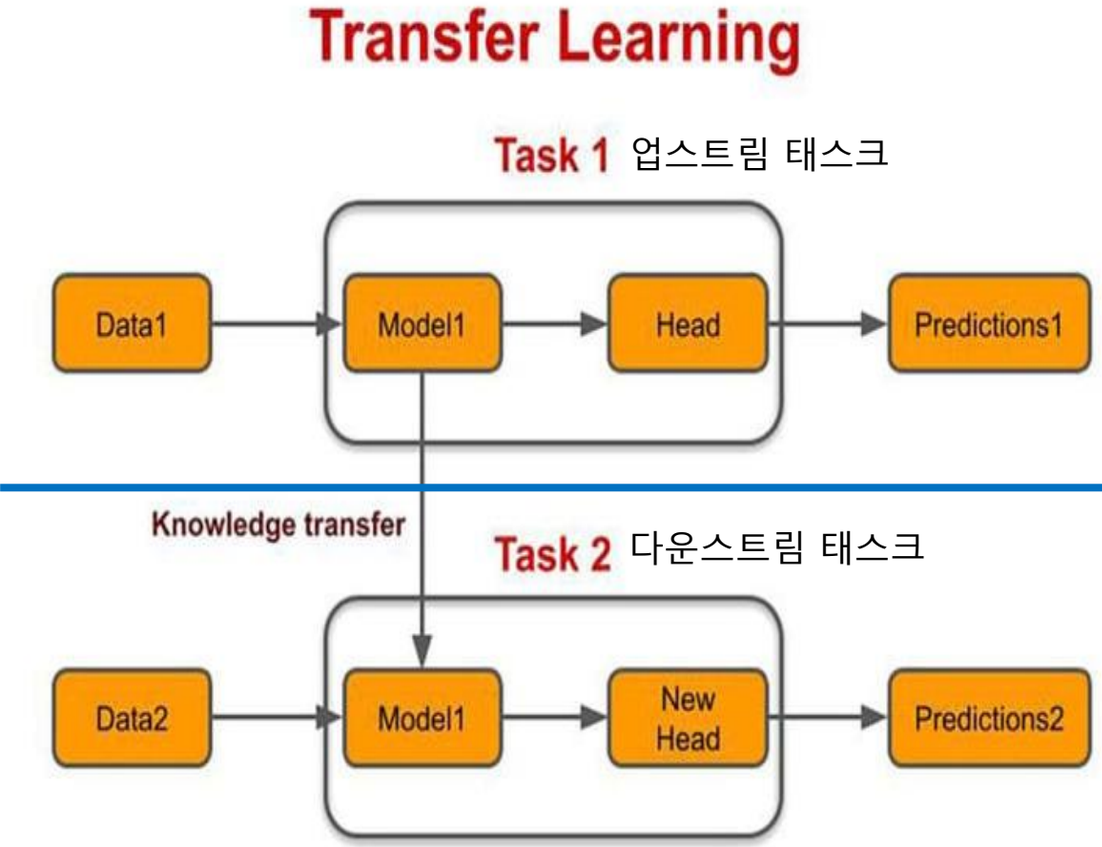


- 트랜스포머 Decoder 해당

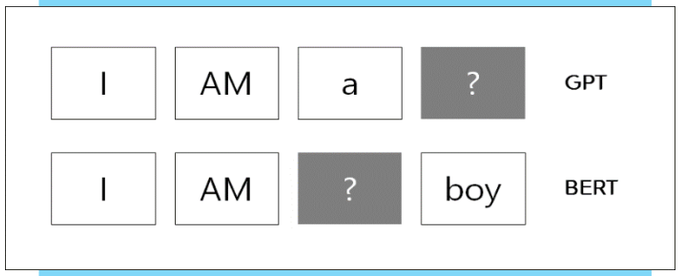
- 적은 데이터로도 뛰어난 성능

- 문장 생성에 뛰어난 성능

자연어 처리 모델 학습 방법



훈련 방법



다음 단어 맞추기 (언어모델)

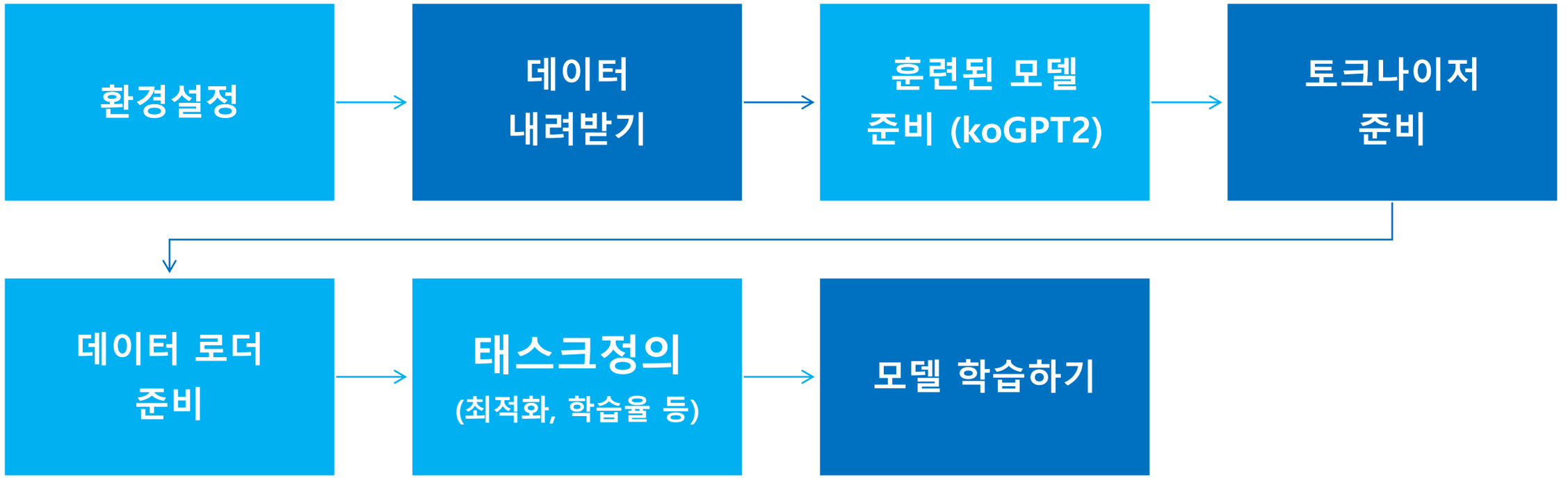
빈칸 채우기 (마스크 언어모델)

분류(classification)

자기 지도 학습(self-supervised learning)

파인튜닝(fine-tuning)
프리트레인을 마친 모델을 다운스트림 태스크 업데이트

학습 파이프라인


환경설정
데이터
내려받기
훈련된 모델
준비 (koGPT2)
토크나이저
준비
데이터 로더
준비
태스크정의
(최적화, 학습율 등)
모델 학습하기

모델과 토크나이저

```
koGPT2_TOKENIZER = PreTrainedTokenizerFast.from_pretrained("skt/kogpt2-base-v2",
                    bos_token=BOS, eos_token=EOS, unk_token='<unk>',
                    pad_token=PAD, mask_token=MASK)
model = GPT2LMHeadModel.from_pretrained('skt/kogpt2-base-v2')

path = 'C:\\Users\\비트캠프\\Desktop\\Team\\'
Chatbot_Data = pd.read_csv(path + "dataset_final.csv")
Chatbot_Data = Chatbot_Data.dropna()
```

```python
class ChatbotDataset(Dataset):
    def __init__(self, chats, max_len=40):
        self._data = chats
        self.max_len = max_len
        self.q_token = Q_TKN
        self.a_token = A_TKN
        self.sent_token = SENT
        self.eos = EOS
        self.mask = MASK
        self.tokenizer = koGPT2_TOKENIZER

    def __len__(self):
        return len(self._data)

    def __getitem__(self, idx):
        turn = self._data.iloc[idx]
        q = turn["Q"]
        q = re.sub(r"([?.!,])", r" ", q)
        a = turn["A"]
        a = re.sub(r"([?.!,])", r" ", a)
        q_toked = self.tokenizer.tokenize(self.q_token + q + self.sent_token)
        q_len = len(q_toked)
        a_toked = self.tokenizer.tokenize(self.a_token + a + self.eos)
        a_len = len(a_toked)

        labels = [self.mask,] * q_len + a_toked[1:]

        mask = [0] * q_len + [1] * a_len + [0] * (self.max_len - q_len - a_len)
        labels_ids = self.tokenizer.convert_tokens_to_ids(labels)

        while len(labels_ids) < self.max_len:
            labels_ids += [self.tokenizer.pad_token_id]


        token_ids = self.tokenizer.convert_tokens_to_ids(q_toked + a_toked)

        while len(token_ids) < self.max_len:
            token_ids += [self.tokenizer.pad_token_id]


        return (token_ids, np.array(mask), labels_ids)

def collate_batch(batch):
    data = [item[0] for item in batch]
    mask = [item[1] for item in batch]
    label = [item[2] for item in batch]
    return torch.LongTensor(data), torch.LongTensor(mask), torch.LongTensor(label)

train_set = ChatbotDataset(Chatbot_Data, max_len=40)

train_dataloader = DataLoader(train_set, batch_size=32, num_workers=0, shuffle=True,
                              collate_fn=collate_batch,)
```

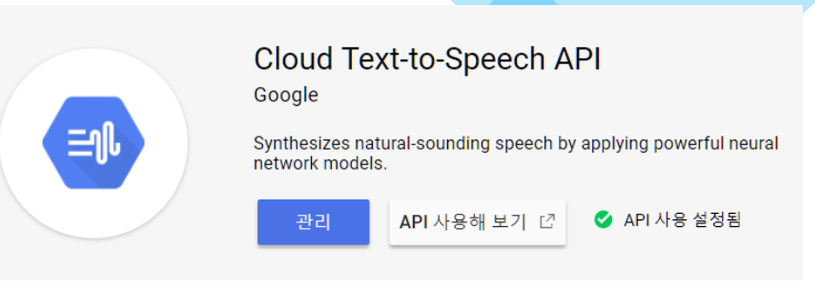
파인 튜닝

```
learning_rate = 3e-5
criterion = torch.nn.CrossEntropyLoss(reduction="none")
optimizer = torch.optim.AdamW(model.parameters(), lr=learning_rate)

epoch = 10 # 학습 횟수
Sneg = -1e18 # 오차가 음수가 되는 것을 방지하기 위한 최소값
```

```
print ("학습 시작")
for epoch in range(epoch):
    for batch_idx, samples in enumerate(train_dataloader):
        optimizer.zero_grad()
        token_ids, mask, label = samples
        out = model(token_ids)
        out = out.logits
        mask_3d = mask.unsqueeze(dim=2).repeat_interleave(repeats=out.shape[2], dim=2)
        mask_out = torch.where(mask_3d == 1, out, Sneg * torch.ones_like(out))
        loss = criterion(mask_out.transpose(2, 1), label)
        avg_loss = loss.sum() / mask.sum()
        avg_loss.backward()
        optimizer.step()
print ("학습 종료")
```

STT - TTS

Google Cloud Speech API



STT / TTS

```python
import os
import winsound
import speech_recognition as sr

os.environ["GOOGLE_APPLICATION_CREDENTIALS"]="chat_api.json"

def say_anything():
    try:
        r = sr.Recognizer()
        # microphone에서 auido source를 생성합니다
        with sr.Microphone() as source:
            print("<<< 마이크에 이야기 하세요 >>>")
            winsound.PlaySound("ns_1_01.wav", winsound.SND_FILENAME)
            audio = r.listen(source)
            answer = r.recognize_google(audio, language='ko')
            return answer
    except:
        print("마이크 입력 에러 입니다")
        return say_anything()
```

STT

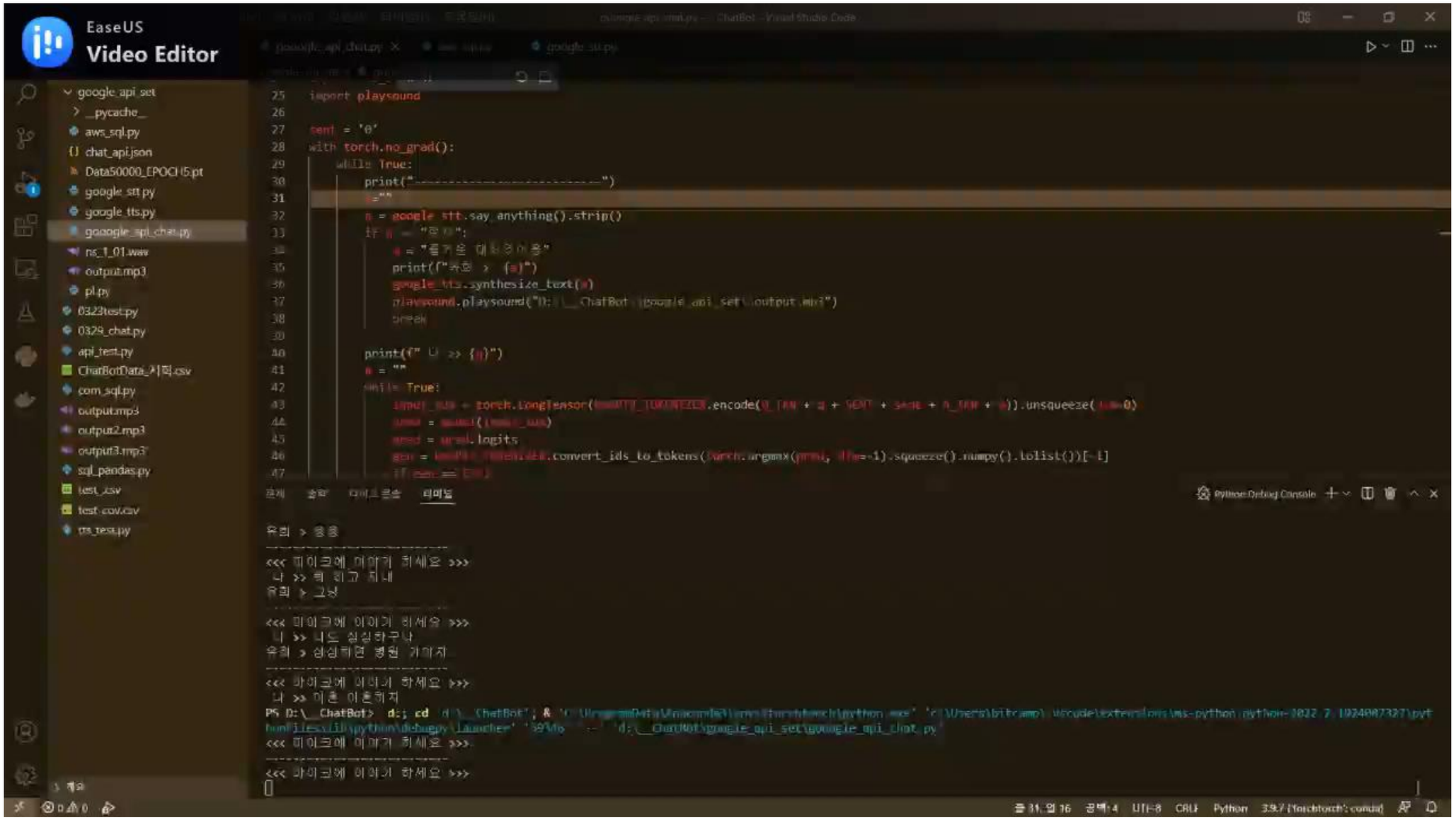
```python
import os
import playsound

def synthesize_text(text):
    os.environ["GOOGLE_APPLICATION_CREDENTIALS"]="D:\\__ChatBot\\google_api_set\\chat_api.json"
    """Synthesizes speech from the input string of text."""
    from google.cloud import texttospeech

    client = texttospeech.TextToSpeechClient()
    input_text = texttospeech.SynthesisInput(text=text)

    # Note: the voice can also be specified by name.
    # Names of voices can be retrieved with client.list_voices().
    voice = texttospeech.VoiceSelectionParams(
        language_code="ko-KR",
        name="ko-KR-Wavenet-A",
        ssml_gender=texttospeech.SsmlVoiceGender.FEMALE,
    )
    audio_config = texttospeech.AudioConfig(
        audio_encoding=texttospeech.AudioEncoding.MP3
    )
    response = client.synthesize_speech(
        request={"input": input_text, "voice": voice, "audio_config": audio_config}
    )
    # The response's audio_content is binary.
    with open("D:\\__ChatBot\\google_api_set\\output.mp3", "wb") as out:
        out.write(response.audio_content)
        # print('Audio content written to file "output.mp3"')
```

TTS

```python
sent = '0'
with torch.no_grad():
    while True:
        print("----------------------------")
        q=""
        q = google_stt.say_anything().strip()
        if q == "잘자":
            a = "즐거운 대화였어용"
            print(f"유희 >  {a}")
            google_tts.synthesize_text(a)
            playsound.playsound("D:\\__ChatBot\\google_api_set\\output.mp3")
            break

        print(f" 나 >> {q}")
        a = ""
        while True:
            input_ids = torch.LongTensor(koGPT2_TOKENIZER.encode(Q_TKN + q + SENT + sent + A_TKN + a)).unsqueeze(dim=0)
            pred = model(input_ids)
            pred = pred.logits
            gen = koGPT2_TOKENIZER.convert_ids_to_tokens(torch.argmax(pred, dim=-1).squeeze().numpy().tolist())[-1]
            if gen == EOS:
                break
            a += gen.replace("_", " ")

        a = a.strip()
        print(f"유희 > {a}")
        google_tts.synthesize_text(a)
        playsound.playsound("D:\\__ChatBot\\google_api_set\\output.mp3")
        aws_sql.insert_QnA(q,a)
        continue
```

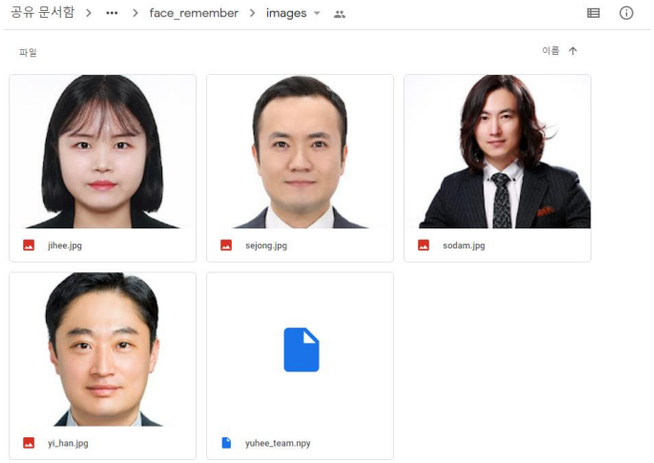
```
<class 'pandas.core.frame.DataFrame'>
                Question                Answer
0     보이스피싱 다 해 본 적 있어요  보이스피싱은 돈을 잃는 경우가 많습니다
2     보이스피싱 다 해 본 적 있어요  보이스피싱은 돈을 잃는 경우가 많습니다
3              만만한게 없네              만만한게 뭐가 있어
4             권리 오겠습니다                  훙훙
..                   ...                 ...
113         너와 나의 연결 고리          연결고리가 뭔지 모르겠엉
114           우리 무슨 사이야    내가 아니라 너님이랑 사이좋게 지내
115  관계를 명확하게 해 줬으면 좋겠어      웅웅      결혼도 안했으면서
116   그래도 상도로 하는게 있는 거야                  웅웅
117        나 말고 딴 남자 만나          만나면 안되니까
```

STT / TTS


EaseUS
Video Editor
google_api_set
__pycache__
aws_sql.py
chat_api.json
Data50000_EPOCH5.pt
google_stt.py
google_tts.py
google_api_chat.py
ns_1_01.wav
output.mp3
pl.py
0323test.py
0329_chat.py
api_test.py
ChatBotData_시작.csv
com_sql.py
output.mp3
output2.mp3
output3.mp3
sql_pandas.py
test_.csv
test-copy.csv
tts_test.py
import playsound
sent = '0'
with torch.no_grad():
while True:
print("~~~~~~~~~~~~~~~~~~~~~~~~~")
a=""
a = google_stt.say_anything().strip()
if a == "끝":
a = "즐거운 대화였어용"
print("유희 > {a}")
google_tts.synthesize_text(a)
playsound.playsound("D:__ChatBot\google_api_set\output.mp3")
break
print(f" 나 >> {a}")
a = ""
while True:
input_ids = torch.LongTensor(
.unsqueeze(
output>
pred = output.logits
gen =
.convert_ids_to_tokens(torch.argmax(pred, dim=-1).squeeze().numpy().tolist())[-1]
if gen ==
문제
출력
디버그콘솔
터미널
Python Debug Console
유희 > 응응
<<< 마이크에 이야기 하세요 >>>
나 >> 뭐 하고 있니
유희 > 그냥
<<< 마이크에 이야기 하세요 >>>
나 >> 너도 심심하구나
유희 > 심심하면 병원 가야지
<<< 마이크에 이야기 하세요 >>>
나 >> 이혼 이혼하지
PS D:__ChatBot> d:; cd 'd:__ChatBot'; & 'C:\Users\bitcamp\.vscode\extensions\ms-python.python-2022.2.1024087327\pyt
hon\lib\python\debugpy\launcher' '59
'd:__ChatBot\google_api_set\google_api_chat.py'
<<< 마이크에 이야기 하세요 >>>
<<< 마이크에 이야기 하세요 >>>

Face Remember

```python
cap = cv2.VideoCapture(0) # 웹캠을 카메라로 사용
cap.set(3,640) # 너비
cap.set(4,480) # 높이

def selfy():
    ret, frame = cap.read() # 사진 촬영
    frame = cv2.flip(frame, 1) # 좌우 대칭

    cv2.imwrite('test_img\my_pic.jpg', frame) # 사진 저장

    cap.release()
    cv2.destroyAllWindows()
    print("찰칵~")

if __name__ =='__main__':
    selfy()
```



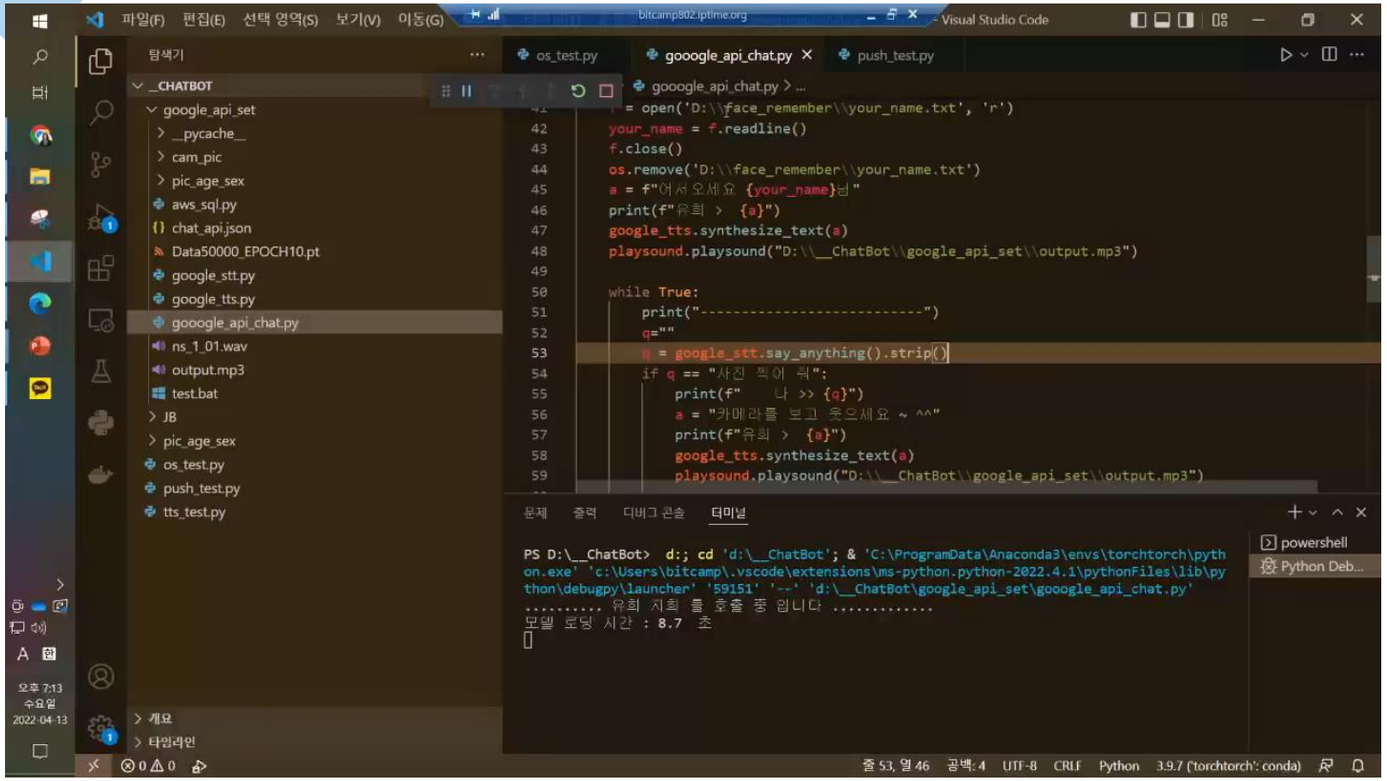
얼굴인식

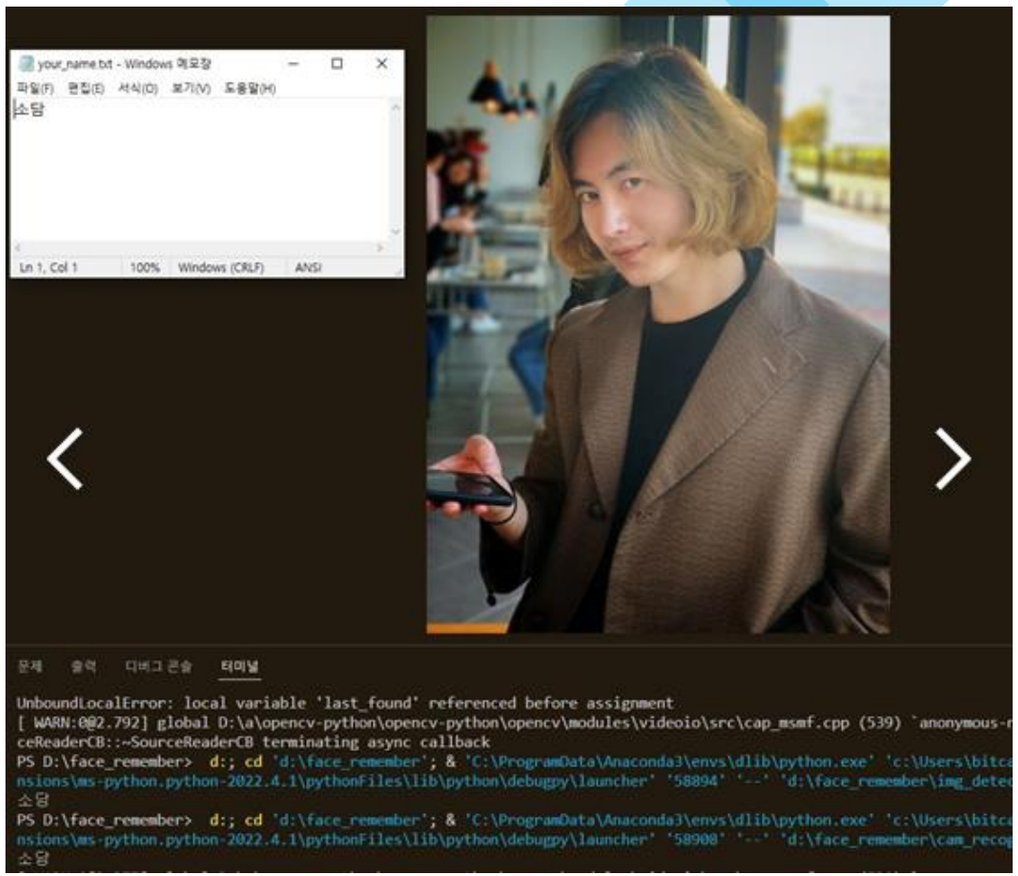
```python
# Compute Saved Face Description
img_paths = {
    '소담': 'images/sodam.jpg',
    '이한': 'images/yi_han.jpg',
    '세종': 'images/sejong.jpg',
    '지희': 'images/jihee.jpg'
}

descs = {
    '소담': None,
    '이한': None,
    '세종': None,
    '지희': None
}

for name, img_paths in img_paths.items():
    img_bgr = cv2.imread(img_paths)
    img_rgb = cv2.cvtColor(img_bgr, cv2.COLOR_BGR2RGB)
    plt.imshow(img_rgb)
    plt.show()

    _, img_shapes, _ = find_faces(img_rgb)

    descs[name] = encode_faces(img_rgb, img_shapes)[0]
    np.save('images/yuhee_team.npy', descs)
print(descs)
```

```python
detector = dlib.get_frontal_face_detector()
sp = dlib.shape_predictor('models/shape_predictor_68_face_landmarks.dat')
facerec = dlib.face_recognition_model_v1('models/dlib_face_recognition_resnet_model_v1.dat')

def find_faces(img):
    dets = detector(img, 1)
    # face Not Found empty 0 return
    if len(dets) == 0:
        return np.empty(0), np.empty(0), np.empty(0)

    rests, shapes = [], []
    shapes_np = np.zeros((len(dets), 68, 2), dtype=np.int)

    for k, d in enumerate(dets):
        rect = ((d.left(), d.top()), (d.right(), d.bottom()))
        rests.append(rect)
        shape = sp(img, d)

        # convert dlib shape to numpy array
        for i in range(0, 68):
            shapes_np[k][i] = (shape.part(i).x, shape.part(i).y)

        shapes.append(shape)
    return rests, shapes, shapes_np

def encode_faces(img, shapes):
    face_descriptors = []
    for shape in shapes:
        face_descriptor = facerec.compute_face_descriptor(img, shape)
        face_descriptors.append(np.array(face_descriptor))
    return np.array(face_descriptors)
```

Face Remember

```python
def face_to_name():
    selfy.selfy()
    # Numpy 로 저장된 얼굴이미지 화일 읽어오기
    descs = np.load('images/yuhee_team.npy', allow_pickle=True)[()]


    #  얼굴인식을 할 이미지를 읽어오기
    img1_path = 'test_img\my_pic.jpg'
    img_bgr = read_img(img1_path)


    img_rgb = cv2.cvtColor(img_bgr, cv2.COLOR_BGR2RGB)
    dets = detector(img_bgr, 1)

    for k, d in enumerate(dets):
        shape = sp(img_rgb, d)
        face_descriptor = facerec.compute_face_descriptor(img_rgb, shape)
        last_found = {'name': 'unknown', 'dist': 0.6, 'color': (0, 0, 255)}

        for name, saved_desc in descs.items():
            dist = np.linalg.norm([face_descriptor] - saved_desc, axis=1)

            if dist < last_found['dist']:
                last_found = {'name': name, 'dist': dist, 'color': (255, 255, 255)}

    your_name = last_found['name']
    print(your_name)
    return your_name
```


Visual Studio Code
os_test.py
goooogle_api_chat.py
push_test.py
_CHATBOT
google_api_set
__pycache__
cam_pic
pic_age_sex
aws_sql.py
chat_api.json
Data50000_EPOCH10.pt
google_stt.py
google_tts.py
goooogle_api_chat.py
ns_1_01.wav
output.mp3
test.bat
JB
pic_age_sex
os_test.py
push_test.py
tts_test.py
f = open('D:\\face_remember\\your_name.txt', 'r')
your_name = f.readline()
f.close()
os.remove('D:\\face_remember\\your_name.txt')
a = f"어서오세요 {your_name}님"
print(f"유희 > {a}")
google_tts.synthesize_text(a)
playsound.playsound("D:__ChatBot\\google_api_set\\output.mp3")
while True:
print("----------")
q=""
q = google_stt.say_anything().strip()
if q == "사진 찍어 줘":
print(f" 나 >> {q}")
a = "카메라를 보고 웃으세요 ~ ^^"
print(f"유희 > {a}")
google_tts.synthesize_text(a)
playsound.playsound("D:__ChatBot\\google_api_set\\output.mp3")
PS D:__ChatBot> d:; cd 'd:__ChatBot'; & 'C:\ProgramData\Anaconda3\envs\torchtorch\python.exe' 'c:\Users\bitcamp\.vscode\extensions\ms-python.python-2022.4.1\pythonFiles\lib\python\debugpy\launcher' '59151' '--' 'd:__ChatBot\google_api_set\goooogle_api_chat.py'
..... 유희 지희 를 호출 중 입니다 .....
모델 로딩 시간 : 8.7 초
powershell
Python Deb...


your_name.txt - Windows 메모장
소담
UnboundLocalError: local variable 'last_found' referenced before assignment
[WARN:0@2.792] global D:\a\opencv-python\opencv-python\opencv\modules\videoio\src\cap_msmf.cpp (539) `anonymous-
ceReaderCB::~SourceReaderCB terminating async callback
PS D:\face_remember> d:; cd 'd:\face_remember'; & 'C:\ProgramData\Anaconda3\envs\dlib\python.exe' 'c:\Users\bitca
nsions\ms-python.python-2022.4.1\pythonFiles\lib\python\debugpy\launcher' '58894' '--' 'd:\face_remember\img_dete
소담
PS D:\face_remember> d:; cd 'd:\face_remember'; & 'C:\ProgramData\Anaconda3\envs\dlib\python.exe' 'c:\Users\bitca
nsions\ms-python.python-2022.4.1\pythonFiles\lib\python\debugpy\launcher' '58900' '--' 'd:\face_remember\cam_recog
소담

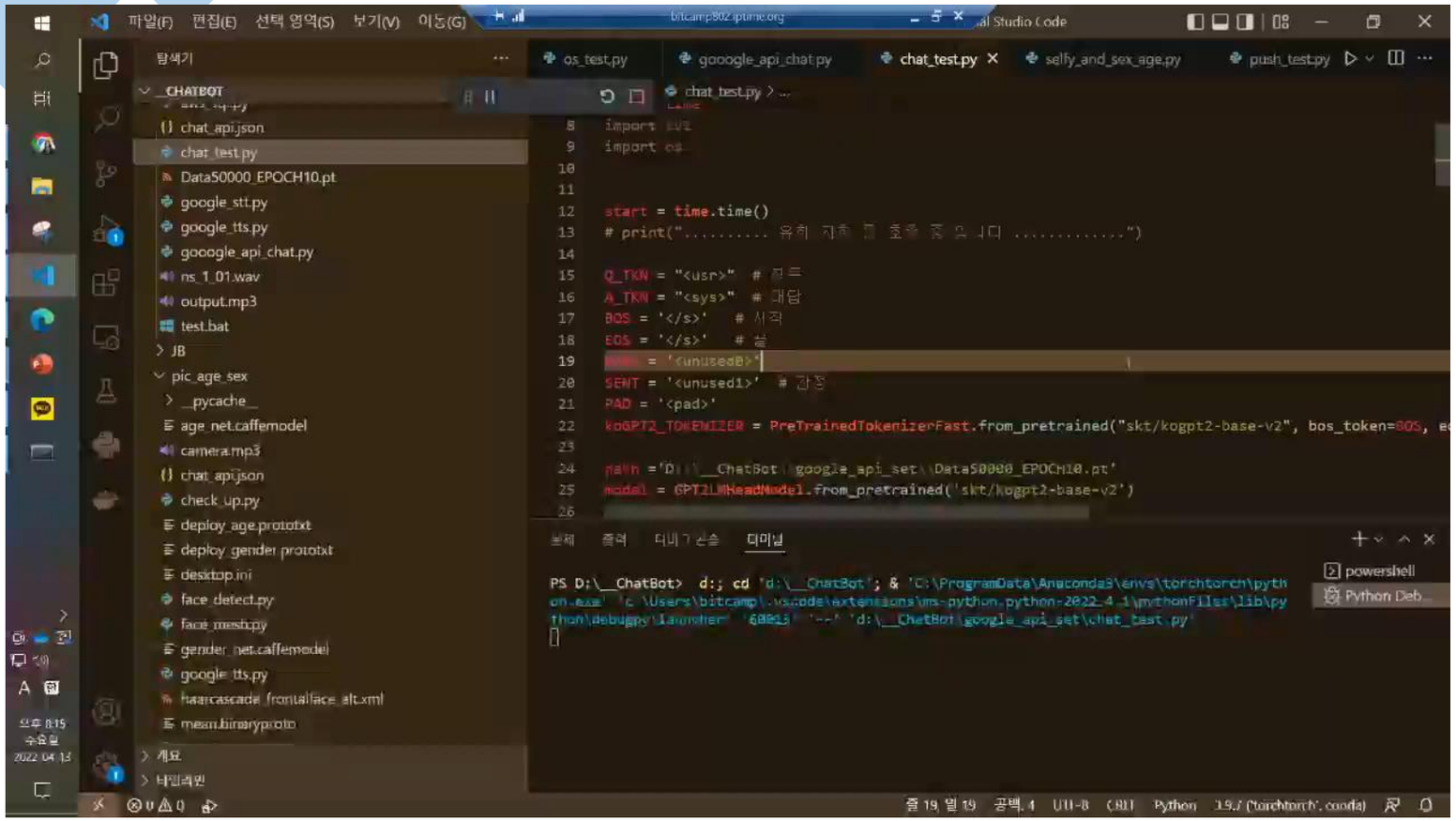
얼굴인식

Face detection

```python
mp_face_detection = mp.solutions.face_detection
mp_drawing = mp.solutions.drawing_utils

# 얼굴 탐지 모델 가중치
cascade_filename = 'haarcascade_frontalface_alt.xml'
# 모델 불러오기
cascade = cv2.CascadeClassifier(cascade_filename)

MODEL_MEAN_VALUES = (78.4263377603, 87.7689143744, 114.895847746)

age_net = cv2.dnn.readNetFromCaffe(
    'deploy_age.prototxt',
    'age_net.caffemodel')

gender_net = cv2.dnn.readNetFromCaffe(
    'deploy_gender.prototxt',
    'gender_net.caffemodel')

age_list = ['0,2','4,6','8,12','15,20','25,32','38,43','48,53','60,100']
gender_list = ['남자', '여자']
```

```python
    # gender detection
    gender_net.setInput(blob)
    gender_preds = gender_net.forward()
    gender = gender_preds.argmax()

    # Predict age
    age_net.setInput(blob)
    age_preds = age_net.forward()
    age = age_preds.argmax()

    age_1 = age_list[age].split(',')[0]
    age_2 = age_list[age].split(',')[1]

    msg = f'당신은 {age_1} 세 에서 {age_2}세 사이의 {gender_list[gender]} 입니다'
    print(msg)
    google_tts.synthesize_text(msg)
    playsound.playsound("output.mp3")
```

나이 & 성별


chat_test.py
os_test.py
gooogle_api_chat.py
selfy_and_sex_age.py
push_test.py
_CHATBOT
chat_api.json
Data50000_EPOCH10.pt
google_stt.py
google_tts.py
gooogle_api_chat.py
ns_1_01.wav
output.mp3
test.bat
JB
pic_age_sex
__pycache__
age_net.caffemodel
camera.mp3
chat_api.json
check_up.py
deploy_age.prototxt
deploy_gender.prototxt
desktop.ini
face_detect.py
face_mesh.py
gender_net.caffemodel
google_tts.py
haarcascade_frontalface_alt.xml
mean.binaryproto
Q_TKN = "<usr>"
A_TKN = "<sys>"
BOS = '</s>'
EOS = '</s>'
SENT = '<unused1>'
PAD = '<pad>'
koGPT2_TOKENIZER = PreTrainedTokenizerFast.from_pretrained("skt/kogpt2-base-v2", bos_token=BOS,
model = GPT2LMHeadModel.from_pretrained('skt/kogpt2-base-v2')
PS D:__ChatBot> d:; cd 'd:__ChatBot'; & 'C:\ProgramData\Anaconda3\envs\torchtorch\python.exe' 'c:\Users\bitcamp\.vscode\extensions\ms-python.python-2022.4.1\pythonFiles\lib\python\debugpy\launcher' '60813' '--' 'd:__ChatBot\google_api_set\chat_test.py'
powershell
Python Deb...


67
68
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERM
nvs\tf270gpu\python.exe' 'c:\Users\bi
내 드라이브\유희지희 프로젝트_yolo\p
INFO: Created TensorFlow Lite XNNPACK
gender Female
age (25 ~ 32)
PS G:\내 드라이브\유희지희 프로젝트_
nvs\tf270gpu\python.exe' 'c:\Users\bi
내 드라이브\유희지희 프로젝트_yolo\p
INFO: Created TensorFlow Lite XNNPACK
gender Female
age (8 ~ 12)

나이 & 성별

Object detection

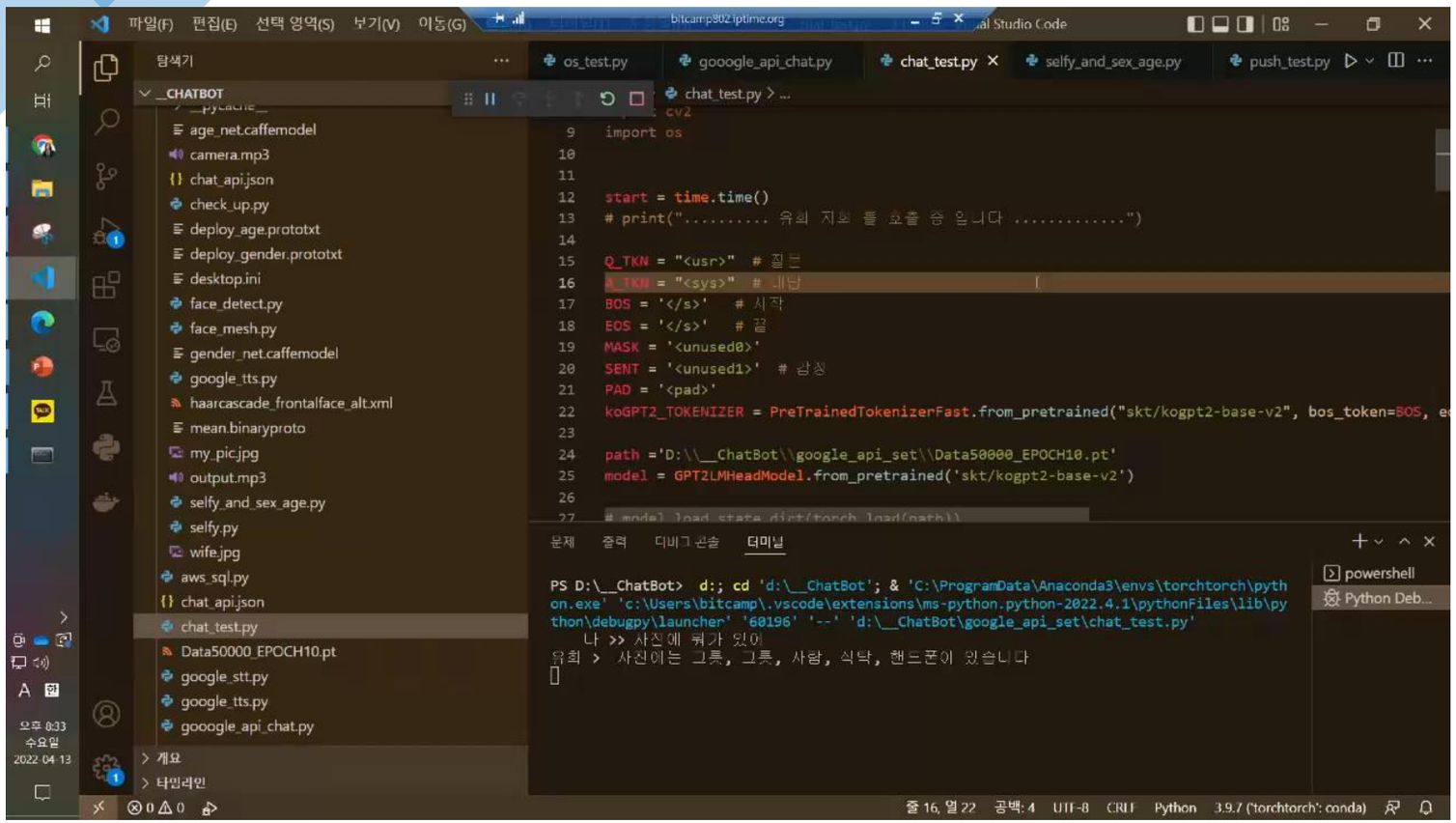
```python
# Load Yolo
net = cv2.dnn.readNet("yolov3.weights", "yolov3.cfg")
classes = []
#클래스 이름을 따로 저장해준다. 이 형식은 클래스가 한글이름 일 때 불러오는 방식이다.
with open("kor_coco.names", "r", encoding='UTF8') as f:
    classes = [line.strip() for line in f.readlines()]
layer_names = net.getLayerNames() # 네트워크의 모든 레이어 이름을 가져옵니다.
output_layers = [layer_names[i - 1] for i in net.getUnconnectedOutLayers()]
# 네트워크의 출력 레이어 이름을 가져옵니다.
colors = np.random.uniform(0, 255, size=(len(classes), 4))

# 이미지 가져오기
img = cv2.imread("wife.jpg")
img = cv2.resize(img, None, fx=0.4, fy=0.4)
height, width, channels = img.shape
```

```python
for out in outs: # 출력을 각각 확인합니다.
    for detection in out: # detection = out[i] = [x, y, w, h, obj_score, class_id]
        scores = detection[5:] # [5:] 는 가장 앞의 5개를 버리고 나머지를 가져옵니다.
        class_id = np.argmax(scores) # 가장 높은 점수를 가진 클래스 아이디를 가져옵니다.
        confidence = scores[class_id]
        if confidence > 0.5: # 확률이 0.5 이상인 것만 가져옵니다.
            # Object detected
            # 탐지된 객체의 너비, 높이 및 중앙 좌표값 찾기
            center_x = int(detection[0] * width)
            center_y = int(detection[1] * height)
            #print(center_x,center_y)
            w = int(detection[2] * width)
            h = int(detection[3] * height)
            #print(w,h)
            # 객체의 사각형 테두리 중 좌상단 좌표값 찾기
            x = int(center_x - w / 2)
            y = int(center_y - h / 2)
            boxes.append([x, y, w, h])
            confidences.append(float(confidence))
            class_ids.append(class_id)

indexes = cv2.dnn.NMSBoxes(boxes, confidences, 0.5, 0.4)
```

사진 분석

Object detection

```
labels = []
for i in range(len(boxes)):
    if i in indexes:
        x, y, w, h = boxes[i] # 객체의 사각형 테두리 중 좌상단 좌표값 찾기
        label = str(classes[class_ids[i]]) # 클래스 이름을 가져옵니다.
        color = colors[i] # 색상을 가져옵니다.
        cv2.rectangle(img, (x, y), (x + w, y + h), color, 2) # 사각형 테두리 그리기
        img = np.array(img) # 이미지를 numpy array로 변환
        # draw.text(img,  label, (x, y + 30), font, 3, color, 3)
        # cv2.putText(img, label, (x, y + 30), font, 3, color, 3) # 텍스트 그리기
        labels.append(label) # 클래스 이름을 리스트에 추가

        b,g,r,a = int(color[0]), int(color[1]), int(color[2]), int(color[3])
        fontpath = "./malgun.ttf"
        font = ImageFont.truetype(fontpath, 32)
        img_pil = Image.fromarray(img)
        draw = ImageDraw.Draw(img_pil)

        draw.text((x , y-50),  label, font = font, fill = (b, g, r, a))
        img = np.array(img_pil)
```


_CHATBOT
age_net.caffemodel
camera.mp3
chat_api.json
check_up.py
deploy_age.prototxt
deploy_gender.prototxt
desktop.ini
face_detect.py
face_mesh.py
gender_net.caffemodel
google_tts.py
haarcascade_frontalface_alt.xml
mean.binaryproto
my_pic.jpg
output.mp3
selfy_and_sex_age.py
selfy.py
wife.jpg
aws_sql.py
chat_api.json
chat_test.py
Data50000_EPOCH10.pt
google_stt.py
google_tts.py
gooogle_api_chat.py
os_test.py
gooogle_api_chat.py
chat_test.py
selfy_and_sex_age.py
push_test.py
import os
start = time.time()
Q_TKN = "<usr>"
A_TKN = "<sys>"
BOS = '</s>'
EOS = '</s>'
MASK = '<unused0>'
SENT = '<unused1>'
PAD = '<pad>'
path ='D:__ChatBot\\google_api_set\\Data50000_EPOCH10.pt'
model = GPT2LMHeadModel.from_pretrained('skt/kogpt2-base-v2')
PS D:__ChatBot> d:; cd 'd:__ChatBot'; & 'C:\ProgramData\Anaconda3\envs\torchtorch\python.exe' 'c:\Users\bitcamp\.vscode\extensions\ms-python.python-2022.4.1\pythonFiles\lib\python\debugpy\launcher' '60196' '--' 'd:__ChatBot\google_api_set\chat_test.py'
나 >> 사진에 뭐가 있어
유희 > 사진에는 그릇, 그릇, 사람, 식탁, 핸드폰이 있습니다


사람
핸드폰
식탁
그릇
그릇

사진 분석

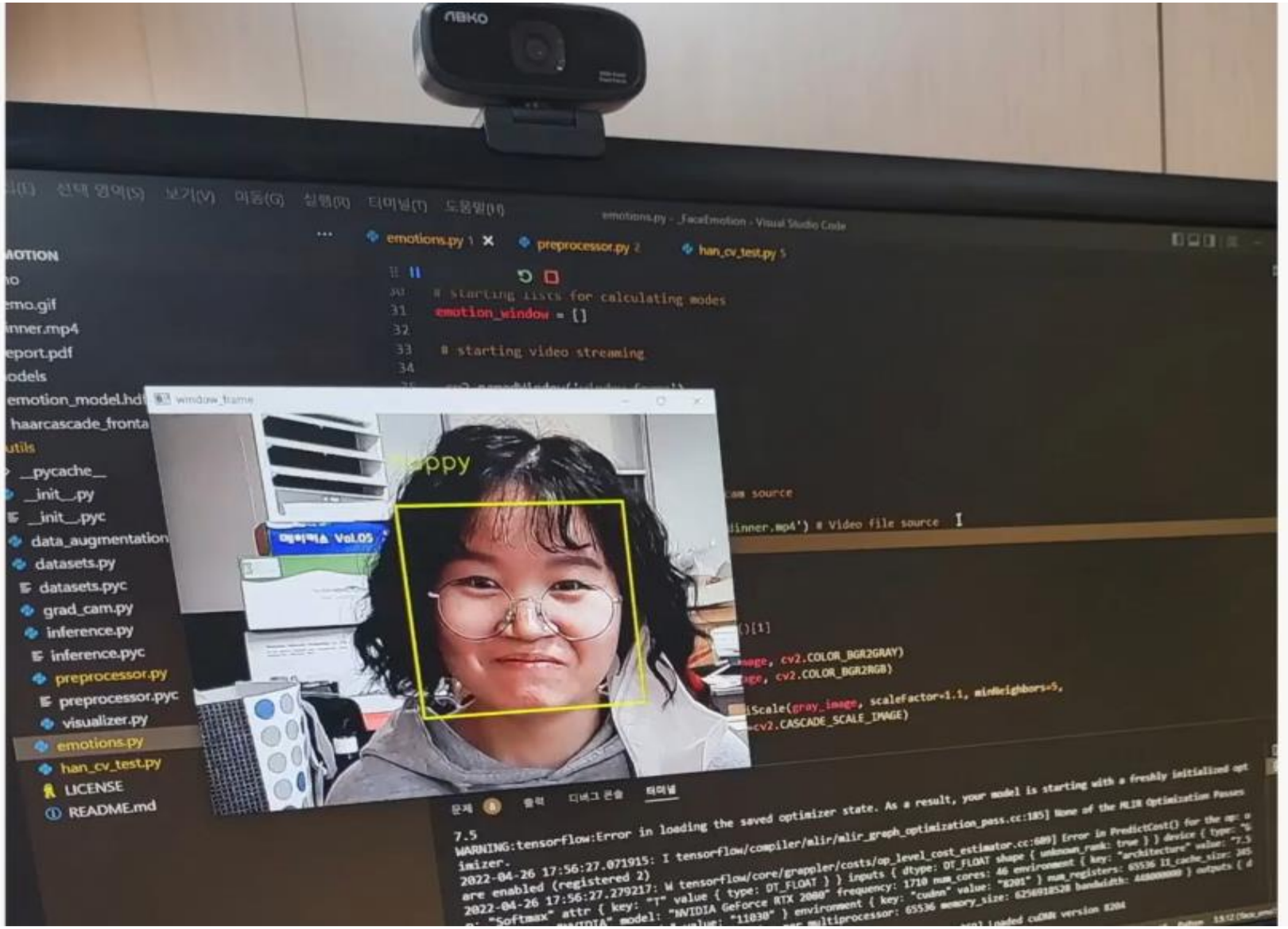
Emotions

```
emotion_model_path = 'C:\\Users\\bitcamp\\Desktop\\projects\\_FaceEmotion\models\\'
emotion_labels = get_labels('fer2013')

# hyper-parameters for bounding boxes shape
frame_window = 10
emotion_offsets = (20, 40)

# loading models
face_cascade = cv2.CascadeClassifier('C:\\Users\\bitcamp\\Desktop\\projects\\_FaceEmotion\\models\\haarcascade_frontalface_default.xml')
emotion_classifier = load_model(emotion_model_path)
```

```
    if emotion_text == 'angry':
        color = emotion_probability * np.asarray((255, 0, 0))
    elif emotion_text == 'sad':
        color = emotion_probability * np.asarray((0, 0, 255))
    elif emotion_text == 'happy':
        color = emotion_probability * np.asarray((255, 255, 0))
    elif emotion_text == 'surprise':
        color = emotion_probability * np.asarray((0, 255, 255))
    else:
        color = emotion_probability * np.asarray((0, 255, 0))

    color = color.astype(int)
    color = color.tolist()

    draw_bounding_box(face_coordinates, rgb_image, color)
    draw_text(face_coordinates, rgb_image, emotion_mode,
                color, 0, -45, 1, 1)

bgr_image = cv2.cvtColor(rgb_image, cv2.COLOR_RGB2BGR)
cv2.imshow('window_frame', bgr_image)
if cv2.waitKey(1) & 0xFF == ord('q'):
    break
```

얼굴 감정 인식

나 오늘 어때 보여



Q&A