CHATBOT HEROES

- KKANGTONG & BAQUI -

이한울 손예선 설진철 한지예

Table of Contents

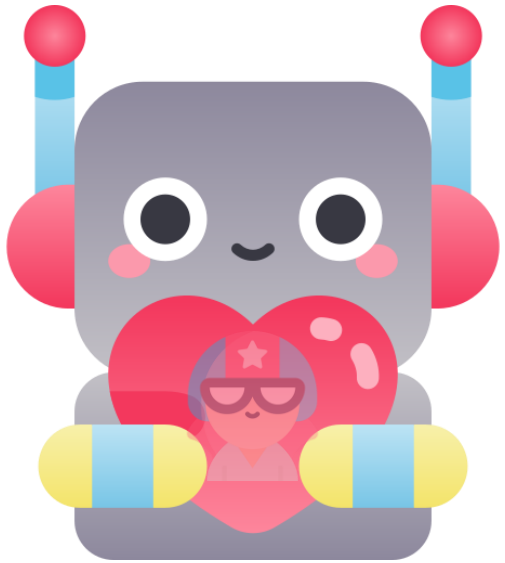
Chatbot Heroes의 장점:

1. 단체채팅기능

- 기존 일상 대화 채팅봇은 오래 대화하면 패턴이 단조롭고 지겨운 경향이 있어, 여러 개의 채팅 모델을 한 번에 로드하여 이를 개선해 보았다.

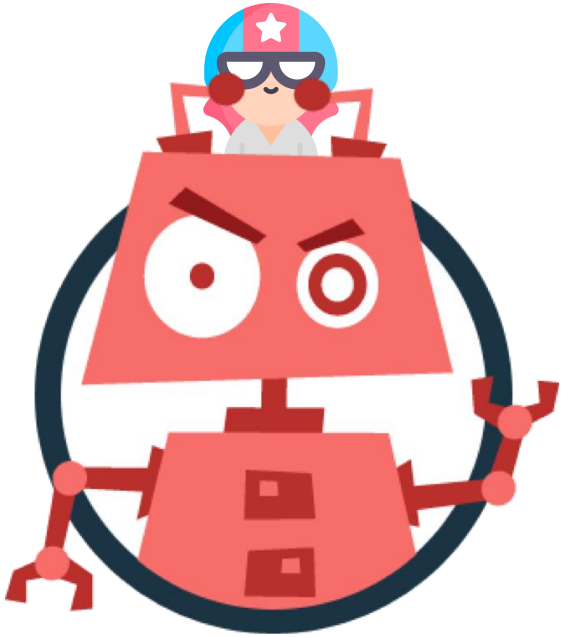
2. 말투에 특징 부여

- 또한, “hgtk Toolkit”를 이용하여 한글자모를 decompose하고 compose 과정을 통해 말투에 변화를 주어 단조로움을 없애 보았다.



- 깡통이 -

- 깡통이는 인간을 좋아합니다.
- 기본 챗봇 대화 데이터를 바탕으로
  정서적으로 좋은 데이터로 훈련될 예정입니다.
- 영어 퀴즈/날씨/미세먼지 기능이 있습니다.



- 바퀴 -

- 바퀴는 영화 매트릭스 광입니다.
- 인간을 정복하고 싶어하지만
  시대를 잘못 타고 태어나서 챗봇이 되었습니다.
- 반말 기능을 추가한 챗봇입니다.
- 상담을 계속 하다보니 까칠해졌습니다.

깡통이 데이터: [Chatbot_Data]와 AI HUB의 [트위터_대화시나리오DB_2000set.xlsx] 데이터를 활용

TraindataMaker.ipynb

```
# 테스트 데이터 정제
kor_pair_test = pd.read_csv('kkangtongData/kor_Pair_test.csv')
kor_pair_test = kor_pair_test[kor_pair_test['is_duplicate']==0]
kor_pair_test_final = kor_pair_test[['question1','question2']]
kor_pair_test_final.rename(columns={'question1':'Q','question2':'A'},
                           inplace = True)
kor_pair_test_final.reset_index(drop = True, inplace = True)
kor_pair_test_final.to_csv('kor_pair_test_final.csv')
```

▲중복데이터 제거

```
import re

for text in data['A']:

    try:
        a = re.findall(r'\w+?님', text)[0]
        #print(a)
        if a[0] != '':
            new = re.sub(a,'',text)
            new.strip()
            #print(new)
            newA.append(new)

    except:
        newA.append(text)
```

▲ 트위터 대화 시나리오 데이터 중 '님'으로 끝나는 호칭들을 정규식을 이용하여 제거

영어 퀴즈를 위해 [초등800단어+예문포함.xlsx] 데이터 활용

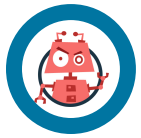
TraindataMaker.ipynb

```
import pandas as pd
df2 = pd.read_excel('[영공카페]+초등800단어+예문포함.xls')
df2.columns = ['word','eng','kor']

## str.split 사용해서 데이터 정제

raw_str = df2.eng[5]
idx = raw_str.find(')')
name = raw_str[1:idx]
context = raw_str[idx+1:]
context.strip()
answer = []
eng2 = []

for i in range(len(df2)):
    raw_str = df2.eng[i]
    idx = raw_str.find(')')
    answer.append(raw_str[1:idx])
    eng2.append(raw_str[idx+1:].lstrip(')').strip('.'))
df2['answer'] = answer
df2['eng2'] = eng2
df2.head()
```

▲ ‘)’와 ‘.’을 strip으로 불필요한 기호 제거

```
quiz = pd.read_csv('Chatbot_data/eng_quiz_data_v2.0.csv')

a = []
for i in quiz['A']:
    a.append(len(i.split(',')))
a = pd.DataFrame(a)
quiz = quiz.drop(0).reset_index(drop = True)

# 문장에서 '_______' 없는 것은 삭제
for i in range(len(quiz['Q'])):
    if len(quiz['Q'][i].split('_______')) < 2:
        print(quiz['Q'][i])
        quiz.drop(i, inplace = True)

quiz.reset_index(inplace = True)
# 검증
for i in range(len(quiz['Q'])):
    if len(quiz['Q'][i].split('_______')) < 2:
        print(quiz['Q'][i])
quiz.drop('index', axis=1,inplace = True)
quiz.to_csv('Chatbot_data/quizfinal.csv')
```

▲ Q&A 형태의 데이터프레임으로 정제



바퀴의 경우, AIHUB의 [한국어_연속적_대화_데이터셋.xlsx]를 활용

TraindataMaker.ipynb

```
baqui = pd.read_excel('kkangtongData/한국어_연속적_대화_데이터셋.xlsx',
                      engine='openpyxl', header=1, usecols='A:C')

baqui.rename(columns={'dialog #':'start','발화':'dialog','감정':'emotion'},inplace = True)
baqui_start = baqui[baqui['start']=='S'] # 발화만 저장
bs_index = baqui_start.index # 발화인덱스

start = []
end = []
for i in bs_index:
    if baqui['emotion'][i+1] in ['분노','혐오']:
        start.append(baqui['dialog'][i])
        end.append(baqui['dialog'][i+1])

df = {'Q':start,'A':end}
df_baqui = pd.DataFrame(data = df)
```

▲첫 발화 데이터만 선택, 그 다음 응답이 부정적인 라벨만을 선택하여 정제

```
start = []
end = []
for i in range(len(baqui.emotion)):
    if baqui.emotion[i+1] in ['분노','혐오']:
        if baqui['start'][i+1] != 'S':
            start.append(baqui.dialog[i])
            end.append(baqui.dialog[i+1])
df = {'Q':start,'A':end}

df_baqui_2 = pd.DataFrame(data = df)
```

▲모든 대화형 데이터 중 응답이 첫 발화가 아닌 데이터에서 부정적인 데이터 선택

Data Augmentation: HANDMADE

TraindataMaker.ipynb

```
baqui_answer = []

for i in range(len(baquiw['Q'])):
    print('{}번째 질문입니다: '.format(i))
    print(baquiw['Q'][i])
    answer = input('바퀴의 답변은?: ')
    if answer = 'quit':
        print('현재까지 완료된 마지막 row는 {}번 입니다.'.format(i))
        baquiw['A'] = baqui_answer
```

- 대략 25,000개인 깡통이 데이터와 비교하여 현저히 적은 수인 약 8,600개의 바퀴 데이터 셋의 증식을 위해 수작업을 진행
- 약 3,000여 건의 부정적인 대화 데이터 셋을 구축

Styling_tone.py

```python
def komoran_token_pos_flat_fn(string):
    tokens_ko = komoran.pos(string)
    pos = [str(pos[0]) + '/' + str(pos[1]) for pos in tokens_ko]
    return pos
```

▲ Komoran 형태소 분석기로 POS Tagging

```python
def exchange_NP(target):
    keyword = []
    ko_sp = komoran_token_pos_flat_fn(target)
    for idx, word in enumerate(ko_sp):
        if word.find('NP') > 0:
            keyword.append(word.split('/'))
            break
    if keyword == []:
        return ''

    if keyword[0][0] == '저':
        keyword[0][0] = '나'
    elif keyword[0][0] == '제':
        keyword[0][0] = '내'
    else:
        return ''

    return keyword[0][0]
```

▲ 존댓말 대명사(NP)인 '저, 제' 를 '나, 내'로 변경

```python
# 종결어미가 아닌 문장 중간의 말투가 변경되는 문제 해결을 위함
# 종결어미의 POS가 JX인 경우, EC가 맨 뒤에 위치하지 않는 경우를 제외
def non_JX(target):
    target = target.strip('.')
    ko_sp = komoran_token_pos_flat_fn(target)[::-1]
    for idx, word in enumerate(ko_sp):
        if (word.find('EC') > 0) & (idx == 0):
            pass
        else:
            target = target[::-1]
            if (word.find('JX') > 0) & (idx == 0):
                target = target.replace(word[0], '', 1)[::-1]
            else:
                target = target[::-1]
    return target
```

▲ 종결어미(EC, EF)가 아닌 문장 중간의 말투가 바뀌는 것을 막기 위해 EC가 맨 뒤에 위치하지 않거나, JX가 맨 뒤에 위치하는 경우를 제외

Styling_tone.py

```python
# 종결어미(EF, EC)만 저장
def make_special_word(target):
    ko_sp = komoran_token_pos_flat_fn(target)[::-1]
    keyword = []

    for idx, word in enumerate(ko_sp):
        if word.find('EF') > 0 :
            keyword.append(word.split('/'))
            _idx = idx
            break

        elif (word.find('EC') > 0) & (idx == 0):
            keyword.append(word.split('/'))
            _idx = idx
            break

        else:
            continue

    if keyword == []:
        return ''

    else:
        keyword = keyword[0]

    return keyword[0]
```

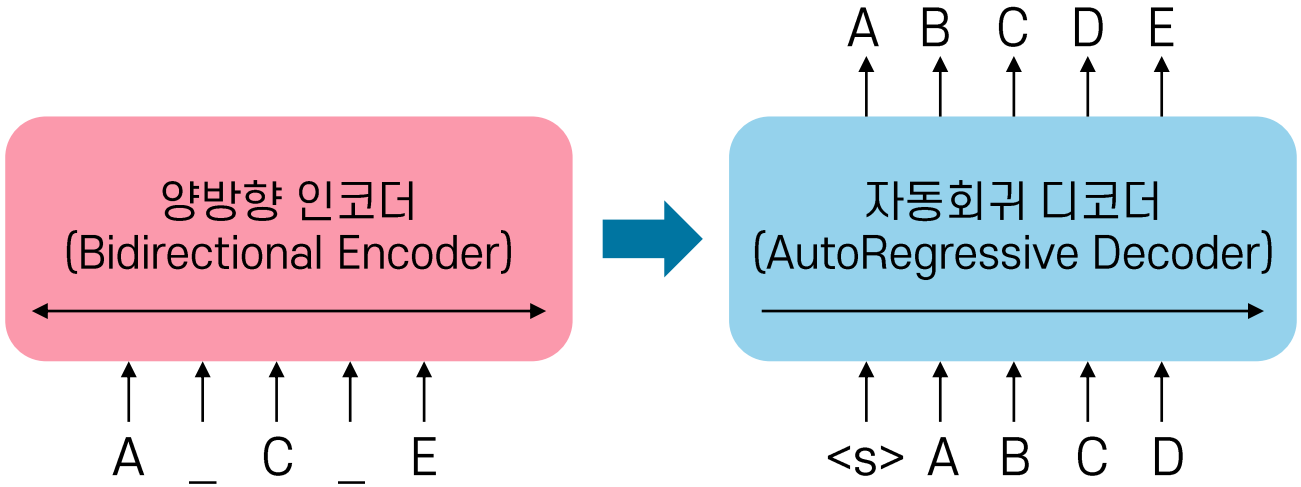
▲ POS Tagging 결과가 종결어미인 경우만 저장

```python
# ~능 말투를 만들어주기 위한 함수
def make_neung(target):
    target = target.rstrip(' ')
    target = target.rstrip(',')
    target = target.rstrip('.')
    hgtk_text = hgtk.text.decompose(target)

    if make_special_word(target) == 'ㄴ가요':
        if target.find('안가요') >= 0:
            target = target.replace(target[target.find('안가요'):], '안가냐능')
        else:
            hgtk_text = hgtk_text.replace(hgtk_text[hgtk_text.find('ㄴᅡㄱᅡᅀᅩ쑈'):], 'ᅀ냐ᅀ느요')
            target = hgtk.text.compose(hgtk_text)

    elif make_special_word(target) == 'ㄴ걸요':
        hgtk_text = hgtk_text.replace(hgtk_text[hgtk_text.find('ㄴᅥㄱᅥㄹᅀᅩ쑈'):], 'ᅀ다ᅀ느요')
        target = hgtk.text.compose(hgtk_text)

    elif make_special_word(target) == 'ㄴ다':
        target = target + '능'

    elif make_special_word(target) == 'ㄴ다고요':
        hgtk_text = hgtk_text.replace(hgtk_text[hgtk_text.find('ㄴᅀ다ᅀㄱᅩᅀᅩ쑈'):], 'ᅀ냐ᅀ느요')
        target = hgtk.text.compose(hgtk_text)

    elif make_special_word(target) == 'ㄴ다니':
        hgtk_text = hgtk_text.replace(hgtk_text[hgtk_text.find('ㄴᅀ다ᅀᄂㅣᅀ'):], 'ᅀ냐ᅀ느요')
        target = hgtk.text.compose(hgtk_text)
```

▲ 챗봇 말투에 차별화를 주기 위해 종결어미에 “~능“ 말투를 적용시켜주는 함수 생성

KoBART 모델

- BART모델의 노이즈 함수(Text Infilling)를 이용하여 한국어 텍스트에 대하여 학습한 한국어 Encoder-Decoder 언어 모델
- BART 모델:
  - Seq2Seq 트랜스포머 구조를 사용했고, ReLU 활성화 함수를 GeLUs로 변경
  - Generation Task에서 사용하기 어렵고, GPT는 Bidirectional한 정보를 얻지 못함
  - 손상된 Text를 입력받아 bidirectional 모델로 인코딩 →
    정답 Text에 대한 likelihood를 자동회귀 디코더로 계산
    ⇒ **noising이 자유로운 장점**
- 대부분 KoBERT나 KoGPT와 같은 모델로 챗봇을 만드는 경우가 많은데, KoBART 모델의 가능성을 보기 위하여 해당 모델을 선택함



KoBART 모델

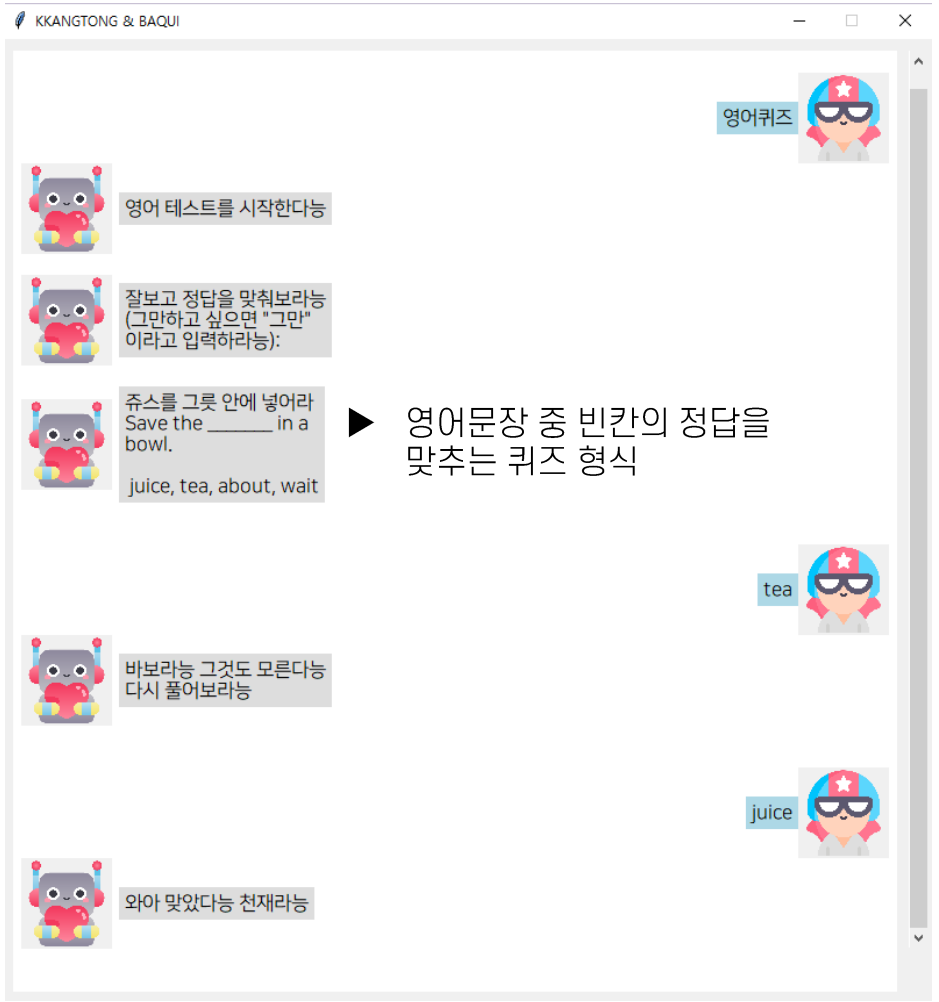
```python
class KoBARTConditionalGeneration(Base):
    def __init__(self, hparams, **kwargs):
        super(KoBARTConditionalGeneration, self).__init__(hparams, **kwargs)
        self.model = BartForConditionalGeneration.from_pretrained(self.hparams.model_path)
        self.model.train()
        self.bos_token = '<s>'
        self.eos_token = '</s>'
        self.tokenizer = PreTrainedTokenizerFast(
            tokenizer_file=os.path.join(self.hparams.tokenizer_path, 'model.json'),
            bos_token=self.bos_token, eos_token=self.eos_token, unk_token='<unk>', pad_token='<pad>', mask_token='<mask>')

    def forward(self, inputs):
        return self.model(input_ids=inputs['input_ids'],
                          attention_mask=inputs['attention_mask'],
                          decoder_input_ids=inputs['decoder_input_ids'],
                          decoder_attention_mask=inputs['decoder_attention_mask'],
                          labels=inputs['labels'], return_dict=True)

    def training_step(self, batch, batch_idx):
        outs = self(batch)
        loss = outs.loss
        self.log('train_loss', loss, prog_bar=True)
        return loss

    def validation_step(self, batch, batch_idx):
        outs = self(batch)
        loss = outs['loss']
        return (loss)

    def validation_epoch_end(self, outputs):
        losses = []
        for loss in outputs:
            losses.append(loss)
        self.log('val_loss', torch.stack(losses).mean(), prog_bar=True)
```

English_teacher.py : 일상대화 챗봇을 일정시간 이용 시 다소 지루해질 수 있다는 문제점을 개선하기 위해 만든 기능

```python
class Englishteacher:

    global Qlen

    def __init__(self,filepath):
        self.filepath = Qfilepath
        self.Question = pd.read_csv(self.filepath)
        self.Answer = pd.read_csv(self.filepath)
        time.sleep(1)
        print('헤헤헤헤')
        time.sleep(0.5)
        Kkangtong("영어테스트를 시작한다능")

    def EnglishtQuestion(self):
        global Qlen
        Question_E = self.Question['Q']
        Qlen = randint(0,len(Question_E))

        return Question_E[Qlen]

    def EnglishAnswer(self):
        global Qlen
        Question_A = self.Question['A']

        return Question_A[Qlen]
```

```python
Kkangtong("채점시간 이라능")
Kkangtong("%2d월%2d일 %02d시%02d분 오늘의 영어점수는 %d 이라능"%(
    now.tm_mon, now.tm_mday, now.tm_hour, now.tm_min,score*10))
```

▼ 채점 기능을 추가해 동기부여



▶ 영어문장 중 빈칸의 정답을 맞추는 퀴즈 형식

Weather_Bot.py

```python
from selenium import webdriver
import re
from bs4 import BeautifulSoup
import unicodedata
import time


def weather_information():
    try:
        driver = webdriver.Chrome("chromedriver.exe")
        time.sleep(1)
        # 네이버 날씨정보

        url = 'https://weather.naver.com/today'
        driver.get(url)
        time.sleep(1)

        # 특정지역 날씨 검색을 위해 검색창 클릭

        searchingarea = driver.find_element_by_css_selector('#header > div.gnb_area > div > div.button_group > button')
        searchingarea.click()
        time.sleep(1)
        # 지역입력
        area = input('지역을 입력하세요(ex 온수동, 청천동): ')
        time.sleep(0.5)
        input_area = driver.find_element_by_css_selector('#_idSearchInput')
        time.sleep(0.5)
        input_area.send_keys(area)
        time.sleep(0.5)
        input_area.submit()
        time.sleep(0.5)

        driver.find_element_by_css_selector('#_idsearchResultContainer > ul > li > a').click()
        time.sleep(0.5)
        # 현재페이지의 정보를 변수에 저장

        html = driver.page_source
        time.sleep(0.5)
        soup = BeautifulSoup(html, 'lxml')
```

▲네이버 날씨 Crawling을 통해 실시간 날씨와 미세먼지 정보 제공

Weather_Bot.py

```python
def Particulate_Matter():

    try:
        driver = webdriver.Chrome("chromedriver.exe")
        time.sleep(1)
        # 네이버 날씨정보

        url = 'https://weather.naver.com/today'
        driver.get(url)
        time.sleep(1)

        # 특정지역 날씨 검색을 위해 검색창 클릭

        searchingarea = driver.find_element_by_css_selector('#header > div.gnb_area > div > div.button_group > button')
        searchingarea.click()
        time.sleep(1)
        # 지역입력

        area = input('지역을 입력하세요(ex 온수동, 청천동): ')
        time.sleep(0.5)
        input_area = driver.find_element_by_css_selector('#_idSearchInput')
        time.sleep(0.5)
        input_area.send_keys(area)
        time.sleep(0.5)
        input_area.submit()
        time.sleep(0.5)

        #input_finalarea = driver.find_element_by_css_selector('#_idsearchResultContainer > ul > li > a')
        #input_finalarea.click()
        driver.find_element_by_css_selector('#_idsearchResultContainer > ul > li > a').click()
        time.sleep(0.5)
        # 현재페이지의 정보를 변수에 저장

        html = driver.page_source
        time.sleep(0.5)
        soup = BeautifulSoup(html, 'lxml')
```

▲네이버 날씨 Crawling을 통해 실시간 날씨와 미세먼지 정보 제공

ChatbotHeroes.py

```
#tkinter GUI 생성
base = Tk()
base.title('KKANGTONG & BAQUI')
base.geometry('800x1000')
base.resizable(False, False)

ChatLog = Text(base, bd=0, bg="white", height="8", width="50", font="Arial",)

ChatLog.config(state=DISABLED)

#위치 태그 지정
ChatLog.tag_configure('tag-right', justify='right')
ChatLog.tag_configure('tag-left', justify='left')

#스크롤바 생성
scrollbar = Scrollbar(base, command=ChatLog.yview, cursor="heart")
ChatLog['yscrollcommand'] = scrollbar.set

#자기소개 버튼 생성
#버튼 클릭 시 자기 소개 출력 하는 intro 함수로 이동
SendButton = Button(base, font=("Verdana",12,'bold'), text="자기소개", width="12", height=5,
                    bd=0, bg="skyblue", activebackground="#3c9d9b",fg='#ffffff', command = intro )

#메세지 입력창 생성
EntryBox = Text(base, bd=0, bg="white",width="29", height="5", font="Arial")

#윈도우 창에 크기 지정 후 배열
scrollbar.place(x=780, y=10, height=772)
ChatLog.place(x=10,y=10, height=810, width=760)
EntryBox.place(x=6, y=830, height=160, width=530)
SendButton.place(x=540, y=830 , height=160, width=250)

#엔터키에 send 함수 바인딩
base.bind_all('<Return>', send)
base.mainloop()
```

▲ tkinter 윈도우 및 GUI 위젯 생성

```
def send(event):

    ChatLog.config(state=NORMAL)
    ChatLog.insert(END, '\n ', 'tag-left')

    msg = message_insert()
    message(msg)

    if msg != '':

        global res1
        global res2

        res1 = baqui_model.chat(msg)
        Baqui(res1)

        res = kkang_model.chat(msg)
        Kkangtong(res)

        ChatLog.insert(END, '\n ', 'tag-right')
        res2 = baqui_model.chat(res)
        Baqui(res2)
```

▲ Enter키 입력 이벤트를 받아 각 훈련 모델에 따른 답변 생성 후, 각 캐릭터 출력 함수로 전달

ChatbotHeroes.py

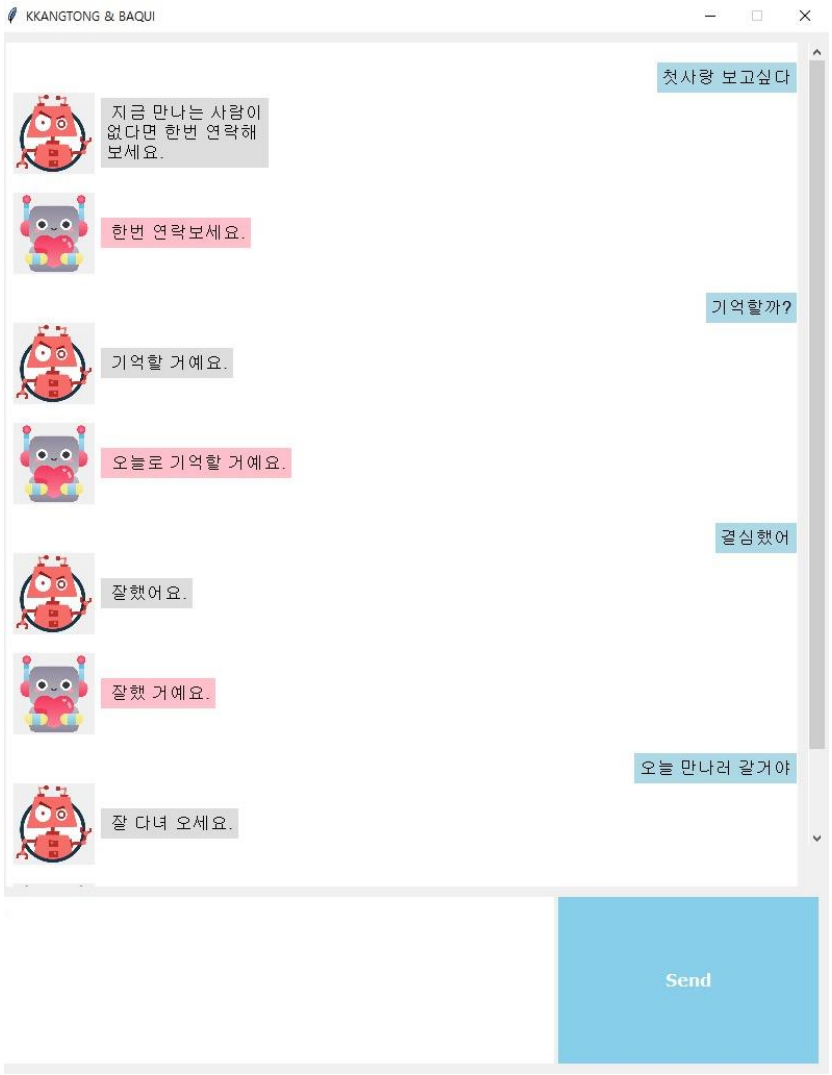
```python
#사용자 메세지 입력
def message_insert():

    #Entry 박스에 입력된 메세지를 가져온 후 입력창 초기화
    msg = EntryBox.get("1.0",'end-1c').strip()
    EntryBox.delete("0.0",END)

    return msg

#사용자 메세지 출력
def message(msg):

    #화면 오른쪽으로 사용자 메세지 정렬
    ChatLog.insert(END,'\n ','tag-right')

    #채팅 윈도우 생성 및 색 지정
    ChatLog.window_create(END, window=Label(ChatLog, fg="#000000", text=msg,
    wraplength=200, font=("Arial", 13), bg="lightblue", bd=4, justify="left"))

    #사용자 프로필 사진 지정
    human_image = tkinter.PhotoImage(file="human.png").subsample(7,7)

    human_label = tkinter.Label(ChatLog, text='인간',image=human_image)

    human_label.image = human_image

    ChatLog.window_create(END, window=human_label)
    ChatLog.insert(END, ' ')

    ChatLog.config(foreground="#442265", font=("Verdana", 12 ))

    ChatLog.yview(END)

    ChatLog.insert(END, '\n ', 'tag-left')
```

▲ 사용자 메시지 입력 받아 채팅창에 출력 하는 함수

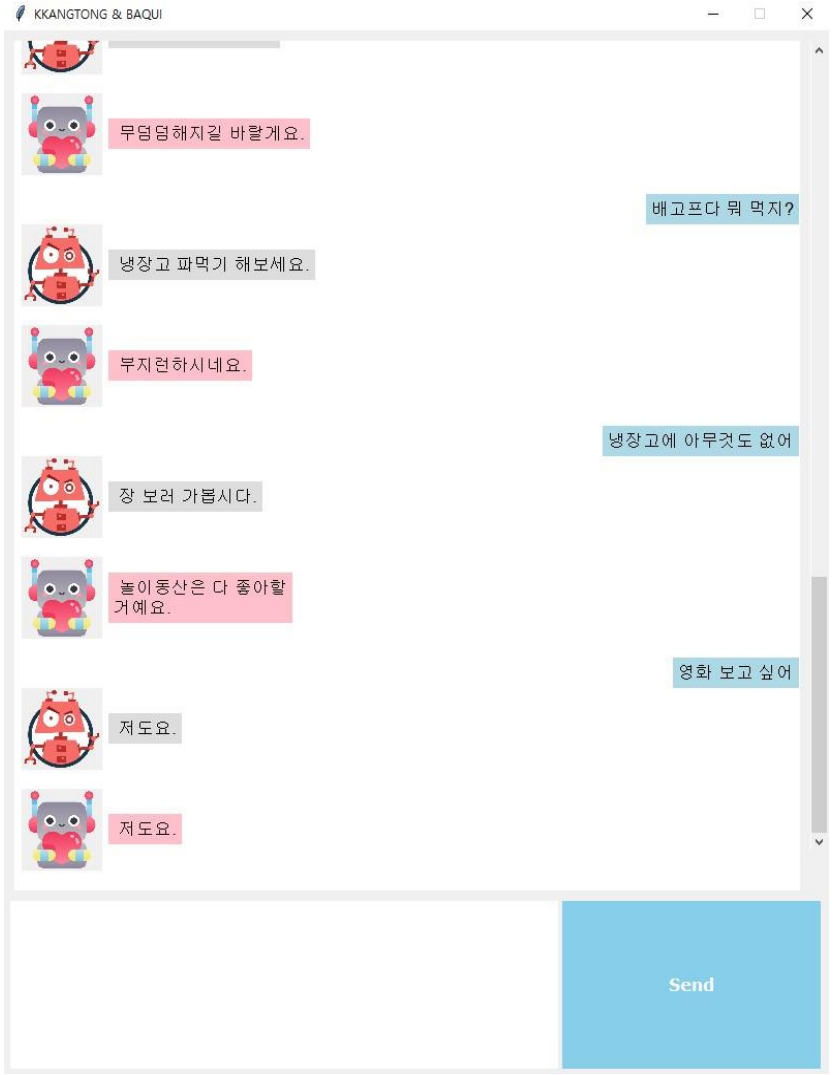
```python
#바퀴 메세지 출력 함수
def Baqui(msg):

    global res1
    global res2

    #대답 대상에 따라 출력 위치 변경
    if msg == res1: #사용자 메세지 대답

        #이미지 생성
        baqui_image = tkinter.PhotoImage(file="robot.png").subsample(7,7)
        baqui_label = tkinter.Label(ChatLog, text='바퀴', image=baqui_image)
        baqui_label.image = baqui_image

        ChatLog.window_create(END, window = baqui_label)
        ChatLog.insert(END, ' ')

        #채팅 윈도우 생성 및 색 지정
        ChatLog.window_create(END, window=Label(ChatLog, fg="#000000", text=msg,
        wraplength=200, font=("Arial", 13), bg="pink", bd=4, justify="left"))

        ChatLog.insert(END, '\n\n ')

        ChatLog.yview(END)

    elif msg == res2: #장릉 메세지 대답

        ChatLog.window_create(END, window=Label(ChatLog, fg="#000000", text=msg,
        wraplength=200, font=("Arial", 13), bg="pink", bd=4, justify="left"))

        baqui_image = tkinter.PhotoImage(file="robot.png").subsample(7,7)
        baqui_label = tkinter.Label(ChatLog, text='바퀴', image=baqui_image)
        baqui_label.image = baqui_image

        ChatLog.window_create(END, window = baqui_label)
        ChatLog.insert(END, ' ')

        ChatLog.insert(END, '\n\n ')

        ChatLog.yview(END)
```

▲ 바퀴 캐릭터 메시지 출력 함수, 대답 대상에 따라 출력 위치 변경

Previous Version

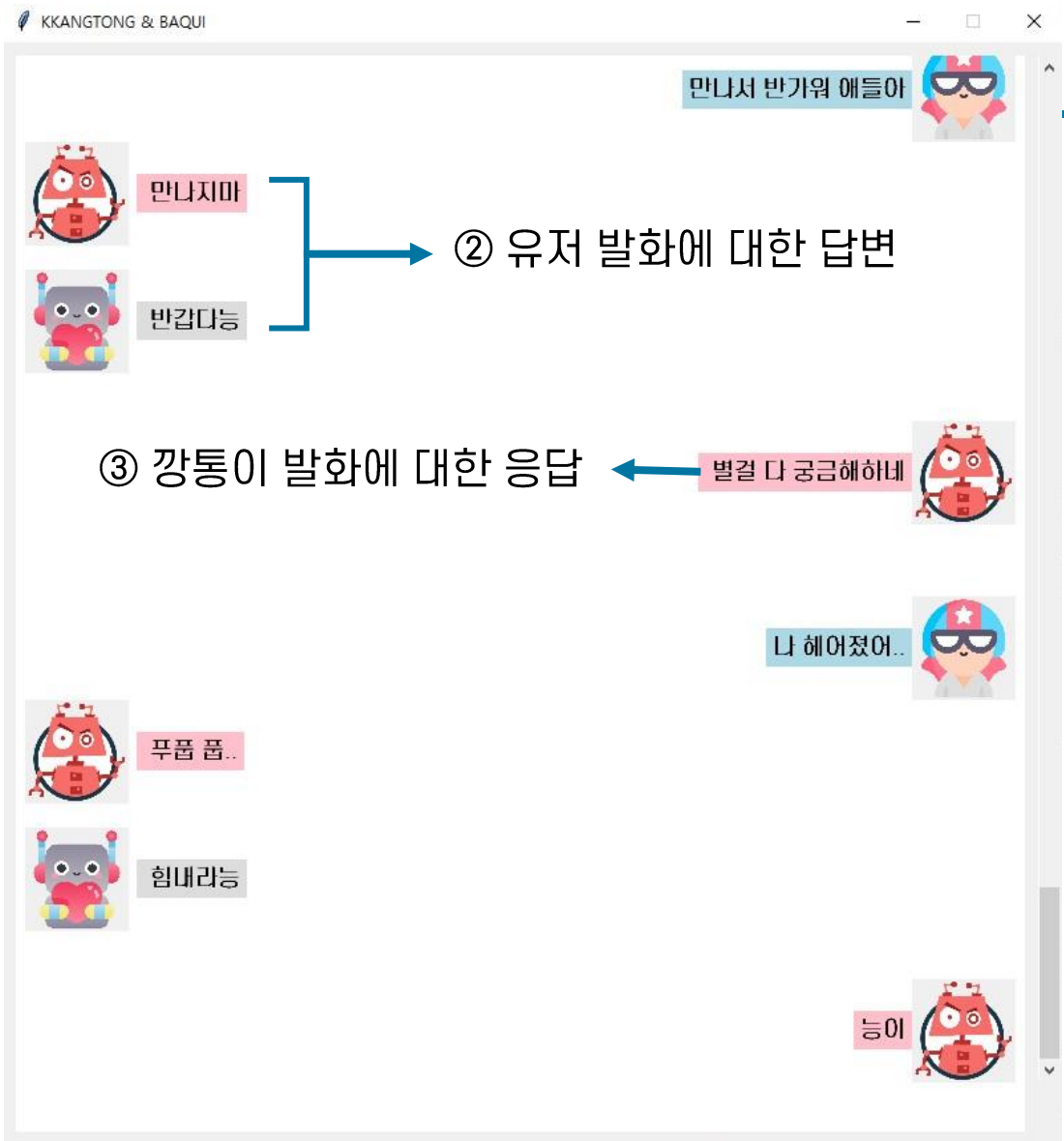


- 연애 상담 대화 -

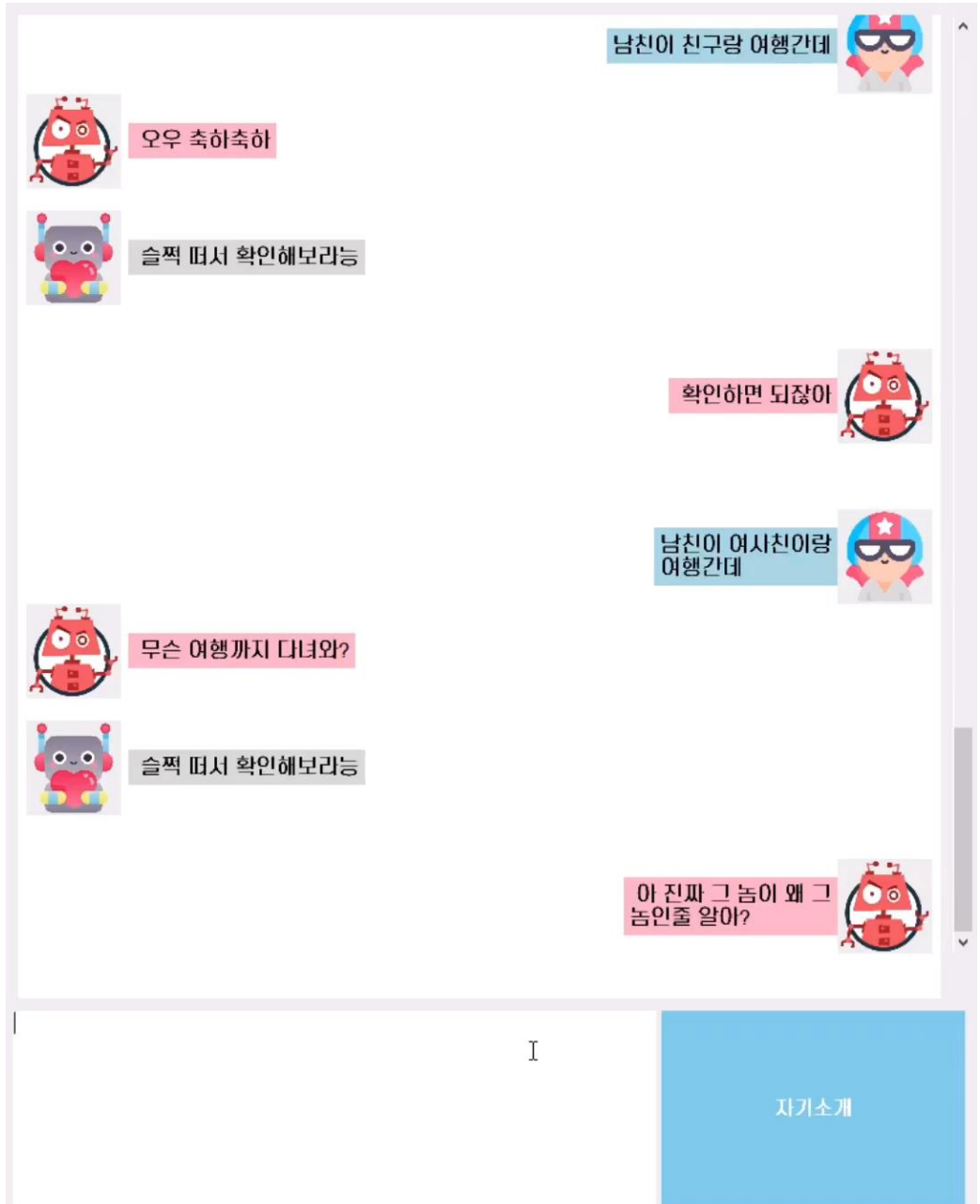


- 일상 대화 -

챗봇히어로즈 채팅 포맷


KKANGTONG & BAQUI
만나서 반가워 애들아
① 유저 발화
만나지마
반갑다능
② 유저 발화에 대한 답변
③ 깡통이 발화에 대한 응답
벌걸 다 궁금해하네
나 헤어졌어..
푸풉 풉..
힘내라능
능이

챗봇히어로즈 시연 영상


남친이 친구랑 여행간데
오우 축하축하
슬쩍 떠서 확인해보라능
확인하면 되잖아
남친이 여사친이랑 여행간데
무슨 여행까지 다녀와?
슬쩍 떠서 확인해보라능
아 진짜 그 놈이 왜 그 놈인줄 알아?
자기소개

References

데이터 출처

- AIHUB 데이터:
  - 트위터에서 수집 및 정제한 대화 시나리오: https://aihub.or.kr/node/269#
  - 한국어 연속적 대화 데이터셋: https://aihub.or.kr/node/273
- 기본 챗봇 데이터: https://github.com/songys/Chatbot_data
- 초등영어 영공카페(초등800단어+예문포함):
  https://m.blog.naver.com/PostView.nhn?blogId=deep_grief&logNo=80184712953&proxyReferer=https:%2F%2Fwww.google.com%2F

참고 자료

- KoBART 모델: https://github.com/SKT-AI/KoBART
- KoBART 챗봇: https://github.com/haven-jeon/KoBART-chatbot
- 감정분석 챗봇: https://github.com/BM-K/Styling-Chatbot-with-Transformer


감사합니다
끝