| **심리케어를 위한 챗봇 서비스** |
| --- |

| **Ⅰ.** |  | **개요** |
| --- | --- | --- |

**1. 배경**

통계청에서 제공한 그래프에 따르면 2021년 한국인의 사망률에 자살이 큰 비중을 차지하고 있다. 특히 10, 20, 30대의 사망원인 1위가 자살이라는 것은 심각한 사안으로 다가오고 있다. 이 서비스는 늘어나고 있는 한국인의 우울증, 정신질환과 같은 심리적 문제를 케어해주기 위한 챗봇 서비스로, 따뜻한 말과 함께 아름다운 사회를 만들어가기 위한 프로젝트이다.

**2. 목표**

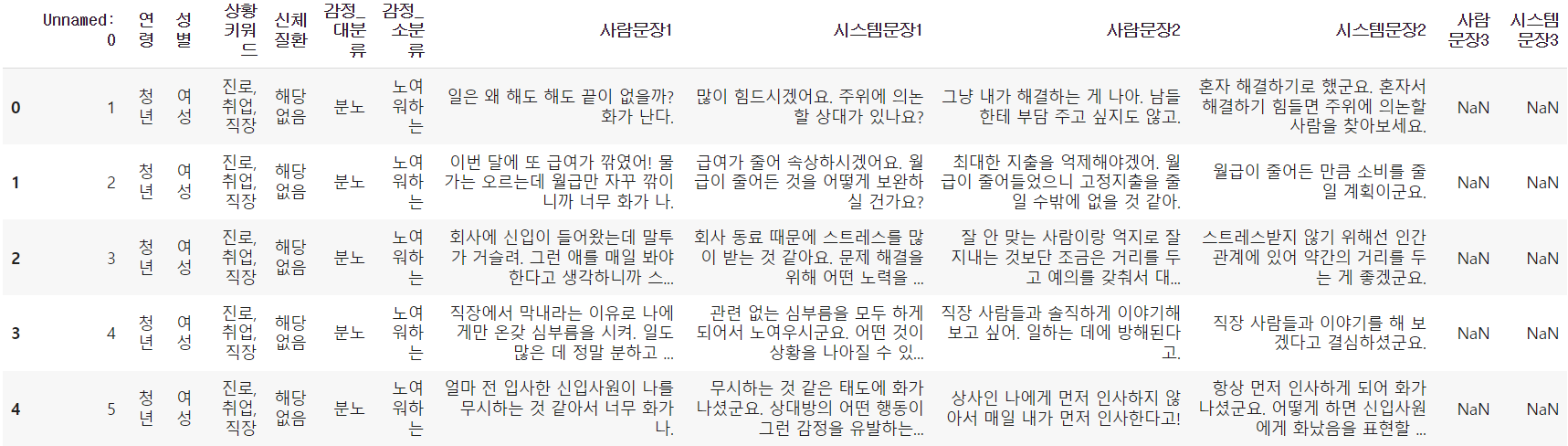
서비스 이용자와의 대화를 통해 감정 상태를 분석하여 감정을 고려한 응답을 생성해낸다.

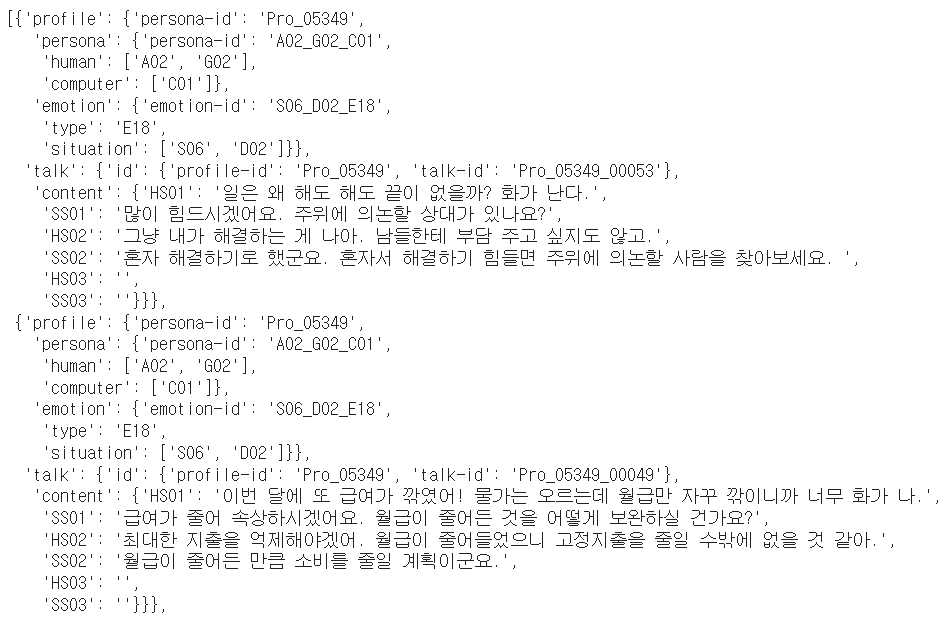
| **Ⅱ.** |  | **문제 수행 내용** |
| --- | --- | --- |

**1. 분석(수행) 절차**

**가. 데이터 수집 및 전처리**

1. 데이터 수집
   1. AI Hub 감성 대화 말뭉치





A -1. 감정 상태 분류표(“emotion” - “type”)

| 대분류  소분류\ | 분노 | 슬픔 | 불안 | 상처 | 당황 | 기쁨 |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | 툴툴대는 | 실망한 | 두려운 | 질투하는 | 고립된 | 감사하는 |
| 2 | 좌절한 | 비통한 | 스트레스 받는 | 배신당한 | 남의 시선 의식하는 | 사랑하는 |
| 3 | 짜증나는 | 후회되는 | 취약한 | 고립된 | 외로운 | 편안한 |
| 4 | 방어적인 | 우울한 | 혼란스러운 | 충격 받은 | 열등감 | 만족스러운 |
| 5 | 안의적인 | 마비된 | 당혹스러운 | 불우한 | 죄책감 | 흥분되는 |
| 6 | 안달하는 | 염세적인 | 회의적인 | 희생된 | 부끄러운 | 느긋한 |
| 7 | 구역질 나는 | 눈물이 나는 | 걱정스러운 | 억울한 | 혐오스러운 | 안도하는 |
| 8 | 노여워 하는 | 낙담한 | 조심스러운 | 괴로워하는 | 한심한 | 신이 난 |
| 9 | 성가신 | 환멸을 느끼는 | 초조한 | 버려진 | 혼란스러운 | 자신하는 |

A - 2. 상황 분류표(“emotion” - “situation”)

| 상황 | 코드 |
| --- | --- |
| 가족관계 | S01 |
| 학업 및 진로 | S02 |
| 학교폭력/따돌림 | S03 |
| 대인관계 | S04 |
| 연애, 결혼, 출산 | S05 |
| 진로, 취업, 직장 | S06 |
| 대인관계(부부, 자녀) | S07 |
| 재정, 은퇴, 노후 준비 | S08 |
| 건강 | S09 |
| 직장, 업무 스트레스 | S10 |
| 건강, 죽음 | S11 |
| 대인관계(노년) | S12 |
| 재정 | S13 |
|

1. 데이터 전처리
   1. 기본 전처리

* 데이터 분할: train\_df1, train\_df2, train\_df3를 생성하여 원본 데이터의 필요한 열을 추출
* 열 이름 변경: 사람문장1-3, 시스템문장1-3 등을 각각 시스템 문장, 사람문장으로 변경
* 결측치 처리: 사람문장, 시스템문장 열에서 NaN 값을 가진 행 삭제
* 데이터 정렬 및 인덱스 재설정: ‘Unnamed: 0’을 기준으로 정렬하고 인덱스를 재설정
  1. 라벨 변환
* 상황키워드와 감정\_대분류에 대한 값 변환: 문자열을 숫자로 변환하여 라벨링
  1. 텍스트 정제
* 한글 텍스트 정제: KoNLPy 라이브러리를 활용하여 텍스트에서 명사, 동사, 형용사 추출
* 중요 단어 추출: TfidVectorizer를 사용하여 ‘사람문장’열의 텍스트를 TF-IDF 벡터로 변환 후 각 문장에서 TF-IDF가 높은 상위 5개의 단어를 추출

**나. 상황 분류 모델**

1. 모델 학습
2. 하이퍼파라미터 설정

* max\_len: 64
* batch\_size: 64
* warmup\_ratio: 0.1
* num\_epochs: 50
* max\_grad\_norm: 1
* log\_interval: 200
* learning\_rate: 5e-5

1. BERTSentenceTransform 클래스 정의

* BERT 스타일의 데이터 변환을 위한 클래스로, 토크나이저, 최대 시퀀스 길이, 어휘 등을 활용하여 입력 문장 또는 문장 쌍을 BERT 모델의 입력 형식으로 변환
* 텍스트 처리: 주어진 입력 ‘line’에서 텍스트를 추출하고, 텍스트 쌍인 경우 ‘text\_a’와 text\_b’로 분리
* 토큰화: 추출한 텍스트를 토그나이저를 사용하여 토큰화 후 ‘tokens\_a’와 ‘tokens\_b’에 토큰 시퀀스를 저장
* 시퀀스 길이 제한: 토큰 시퀀스의 길이를 최대 시퀀스 길이에 맞도록 제한. 문장 쌍인 경우 [CLS], [SEP], [SEP] 토큰에 해당하는 부분을 고려하여 길이를 조절
* 어휘 매핑: 어휘를 사용하여 토큰 ID로 매핑. [CLS] 및 [SEP]토큰을 추가하고, segment\_ids 배열을 초기화
* 패딩 적용: 패딩을 적용하여 시퀀스 길이를 최대 시퀀스 길이에 맞춤
* 배열 반환: 변환된 입력 토큰 ID, 유효 길이, 그리고 입력 토큰 타입 ID를 각각 반환

1. BERTDataset 클래스 정의

* PyTorch의 ‘Dataset’ 클래스를 상속하여 훈련 및 검증 데이터셋을 구성하는 클래스로, 주어진 데이터셋을 BERT 모델에 적용할 수 있는 형태로 반환
* 문장 및 레이블 추출: 주어진 데이터셋에서 텍스트 및 레이블 정보를 추출
* BERTSentenceTransform 적용: ‘BERTSentenceTransform’ 클래스를 사용하여 텍스트를 BERT 모델의 입력 형식으로 변환
* 훈련 및 검증 데이터셋 구성: 변환된 문장 정보와 해당하는 레이블을 리스트에 저장하여 PyTorch 데이터셋을 구성
  1. BERTClassifier 클래스 정의
* Pytorch의 ‘nn.Module’을 상속한 BERT 기반 분류 모델을 정의하는 클래스
* BERT 모델 적용: ‘KoBERTTokenizer’ 및 ‘BertModel’을 사용하여 BERT 모델을 구성
* Hidden Size 및 클래스 수 설정, 드롭아웃 적용
* 선형 분류기 설정: ‘nn.Linear’를 사용하여 BERT의 출력을 주어진 클래스 수에 대한 로짓으로 변환
* Attention Mask 생성: ‘gen\_attention\_mask 메서드를 사용하여 어텐션 마스크를 생성. 어텐션 마스크는 유효한 토큰 길이에 따라 1로 채워지고, 나머지는 0으로 채워짐
* 순방향 전파: ‘forward’ 메서드를 통해 주어진 입력에 대한 순방향 전파를 수행. BERT 모델의 출력에 드롭아웃이 적용된 후, 선형 분류기를 통해 클래스별 로짓을 생성
  1. 모델 학습
* 모델 및 옵티마이저 설정: BERT 기반 분류 모델 생성, AdamW 옵티마이저 사용하여 가중치 감소를 적용하여 모델 초기화
* 코사인 함수와 웜업을 활용한 동적 학습률 조정
* 훈련 중 early stopping을 통해 성능 향상 확인: 검증 정확도가 이전 최고 정확도를 갱신하면 모델 저장
* 훈련과 검증 단계: 손실 및 정확도를 주기적으로 출력하여 학습 진행을 모니터링
* 성능 확인: 21 epoch에서 train acc 0.9670, test acc 0.6746의 성능 확인

**다. 감정 분류 모델**

1. 모델 학습
2. 하이퍼파라미터 설정

* epochs: 10
* patience(검증 손실이 개선되지 않는 에폭 수): 3

1. EmotionDataset 클래스 정의

* PyTorch의 ‘Dataset’ 클래스를 상속하여 주어진 데이터셋을 BERT 모델에 적용할 수 있는 형태의 데이터셋으로 직접 정의하는 클래스
* 주어진 인덱스에 해당하는 입력 시퀀스의 ID, attention mask, 그리고 해당하는 레이블을 반환
* 이 클래스와 DataLoader을 함께 사용하여 모델 학습에 필요한 미니배치 생성

1. JointDistilBertModel 클래스 정의

* ‘nn.Module’을 상속하며, 의도 분류 모델에 자주 사용되는 PyToch의 DistilBERT 모델을 정의하는 클래스
* BERT 모델 적용: 의도 분류를 위해 사전 학습된 ‘DistilBERT’을 사용하여 BERT 모델을 구성
* 클래스 수 설정, 드롭아웃 적용
* 선형 분류기 설정: ‘nn.Linear’를 사용하여 BERT의 출력을 주어진 클래스 수에 대한 로짓으로 변환
* Attention Mask 생성: ‘attention\_mask 메서드를 사용하여 어텐션 마스크를 생성.
* 순방향 전파: ‘forward’ 메서드를 통해 주어진 입력에 대한 순방향 전파를 수행. BERT 모델의 출력에 드롭아웃이 적용된 후, 선형 분류기를 통해 클래스별 로짓을 생성

1. comput\_loss 함수 정의

* 다중 클래스 분류 문제에 적합한 크로스 엔트로피 손실을 이용하여 모델 예측과 실제 레이블 간의 차이를 계산하는 함수

1. 모델 학습

* 모델 및 옵티마이저 설정: 직접 정의한 클래스와 함수를 사용하여 BERT 기반 분류 모델 생성
* 학습 루프: 각 에폭마다 모델을 학습 데이터로 학습시킴. 각 미니 배치를 사용하여 손실을 계산하고, 손실에 대한 역전파를 수행하여 모델을 업데이트
* 조기 종료: 검정 손실이 ‘patience’만큼 개선되지 않으면 학습을 조기 종료하고 해당 에폭에서 훈련을 멈춤
* 훈련과 검증 단계: 손실 및 정확도를 주기적으로 출력하여 학습 진행을 모니터링
* 성능 확인: 6 epoch에서 train loss: 1.79, test loss 1.81의 성능 확인

**라. 응답 생성**

1. 대화 입력 및 전처리

* (대화 입력 예시) “남들은 결혼 전에 일억을 모았다는데 난 뭐를 한 것인지 모르겠어. 자괴감만 드네”
* 명사 추출 함수를 사용하여 입력 문장에서 명사를 추출하고, 이를 공백으로 구분된 문자열로 반환

1. 대화의 감정과 상황 예측

* 입력된 대화를 훈련시켜둔 감정 분류 모델과 상황 분류 모델에 넣어서 ‘감정’과 ‘상황’을 각각 예측
* ‘감정-상황’을 합쳐서 라벨링 할 때보다 따로 라벨링하여 학습했을 때 성능이 향상되었음

1. 대답 반환

대답 데이터 셋에서,

* 동일한 감정
* 동일한 상황 에 해당하는 행들을 필터링하고,

전처리된 입력 문장과 train\_df의 ‘사람문장\_명사’간의 코사인 유사도를 계산. 계산된 코사인 유사도가 가장 높은 행의 시스템 응답을 선택하여 반환

(대답 반환 예시) “결혼 하면 어떤 점에서 그렇게 느끼게 만드나요?”