상담을 위한 사용자의 답변 적극성과 일관성 판단 및 발화 생성 서비스

설계서

(요구기능: 질문에 답변하는 사용자의 일관성 판단)

(요구기능 ID: CP_002)

(참조 파일명: 없음)

문서번호 :

Version 0.1

개정 이력

제.개정내역

버전	승인일자	개요	작성자

배포이력

버전	배포일자	배포처

검토이력

버전	검토일자	검토방법	검토자

목 차

1. 요구기능 설명

제공하는 주요 상세 기능

1. 일관성 평가: 사용자가 답변할 때, 얼마나 일관성을 가지고 질문에 답변하는지 평가한다.

기능의 동작 원리

1. 자연어 처리 (NLP) 기법 활용: 이 기능은 사용자의 답변을 분석하기 위해 자연어 처리 기술을 사용한다. 특히, 답변을 함으로써 이전의 대화를 분석하여 일관성을 파악한다.

2. 일관성 모델링: 사용자의 답변을 분석하여 일관성을 분석한다. 이는 기존의 대화와 일관성 분석, 키워드 일치 여부, 문맥적 정확성을 포함한 다양한 요소를 기반으로 한다.

3. 기계 학습: 시간이 지남에 따라 수집되는 데이터를 통해 모델을 지속적으로 학습시켜, 판단 기준을 최적화하고 정확도를 개선한다.

동작 조건

1. 데이터 입력: 사용자의 답변이 음성 형식으로 입력되어야 하며, 음성 데이터를 텍스트로 변환에 정확성이 필요하다.

2. 정확한 NLP 도구 선택: 한국어에 적합한 NLP 도구와 알고리즘이 필요하다.

주요 처리 대상

1. 사용자 답변: 사용자가 입력한 음성 데이터와 이를 텍스트로 변환한 데이터

2. 질문 데이터: 상담 질문의 내용 및 의도가 담긴 데이터

처리 결과

1. 일관성 판단 결과: 각 사용자 답변에 대한 적극성을 수치로 표현한 결과. 이 점수는 답변의 관련성 및 완전성을 반영한다.

1.1. 주요 상세기능 설명

- ID: FJ_001

- 정의: 주요 상세 기능 #1 은 사용자의 답변 일관성을 평가하는 데 초점을 맞춘다. 이 기능은 사용자의 대답과 이전에 대화했던 데이터를 기반으로 현재의 대답이 이전의 데이터들과 일치하는지 여부를 판단한다.

- 상세 설명: 주요 상세 기능 #1 은 사용자의 답변에 나타나는 일관성을 분석하는 다양한 요소를 평가한다. 이는 현재 답변 키워드, 이전의 답변, 답변을 통해 생성된 화자 정보를 포함한다. 이러한 측정을 통해, 사용자가 주어진 질문에 얼마나 일관적으로 대응하는지를 평가하며, 이를 기반으로 현재 대답의 일관성을 파악한다.

1.2. 요구기능 동작 절차

단계 1. 질문을 통해 얻은 사용자의 답변을 바탕으로 음성 데이터 수집

단계 2. 데이터를 분석하여 일관성 판단

[단계별 상세 내용]

단계 1: 사용자는 시스템을 통해 상담 질문에 답변을 한다. 이 단계에서 시스템은 사용자의 답변을 수집하고 저장한다.

단계 2: 시스템은 수집된 답변을 분석하여 사용자가 얼마나 일관적으로 질문에 응답했는지를 평가한다. 이 평가는 현재 답변 키워드, 이전의 답변, 답변을 통해 생성된 화자 정보 등을 포함한 다양한 기준에 따라 이루어진다. 평가 결과는 현 답변의 일관성 여부로 표현된다.

1.3. 동작 규칙

동작 규칙 1: 데이터 보안과 프라이버시 준수

사용자로부터 수집된 모든 데이터는 데이터 보호 규정과 프라이버시 정책에 따라 처리되어야 한다. 이는 사용자의 신뢰를 유지하고 법적 문제를 예방하는 데 중요하다. 데이터는 암호화되어 안전하게 저장되며, 오직 승인된 시스템과 인력만이 이에 접근할 수 있어야 한다.

동작 규칙 2: 적응형 학습 절차의 적용

시스템은 수집된 데이터를 기반으로 지속적인 학습을 수행하며, 이를 통해 평가 기준을 자동으로 조정할 수 있어야 한다. 이는 머신 러닝 알고리즘의 적응형 학습 기능을 활용하여 이루어지며, 다양한 사용자 행동과 반응을 학습함으로써 시스템의 정확성과 유연성을 강화한다.

1.4. 가정(Assumptions)

- 상담은 정보보호수집 동의서에 동의를 거쳐야 한다.

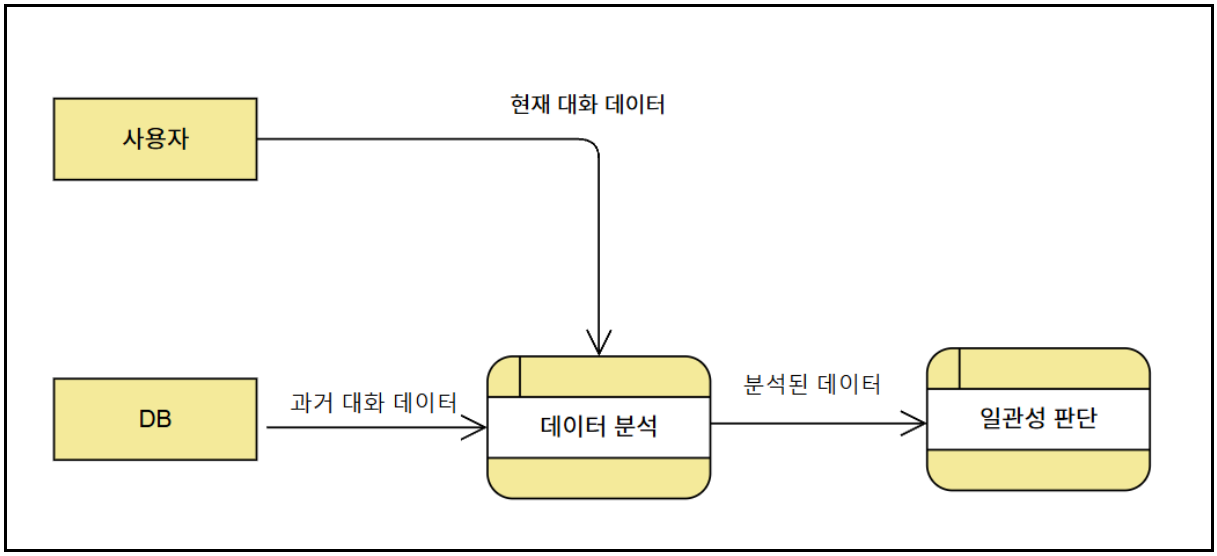
- 사용자가 상담을 거절할 경우 상담을 유도하는 것이 아닌 상담을 종료한다.

- 음성 데이터의 텍스트화가 원할하게 진행되어야 한다.

- 음성 데이터의 개인 정보가 유출되지 않게 잘 보호되어야 한다.

2. 기술설계 #1

2.1. 기술 설계



1. 사용자의 음성 데이터 저장 및 처리

사용자의 음성 데이터와 해당 데이터를 텍스트화하고 저장하는 행위는 모두 CP_001 에서 이루어진다. 그렇기에 해당 코드를 생략한다.

2. 대화 데이터를 DB에 저장

```
@app.route('/upload_conversation', methods=['POST'])
def upload_conversation():
    user_id = request.form['user_id']
    conversation = request.form['conversation']
    timestamp = datetime.now().strftime('%Y-%m-%d %H:%M:%S')

    conn = sqlite3.connect('conversation_data.db')
    cursor = conn.cursor()
    cursor.execute('''INSERT INTO ConversationData (user_id, timestamp, conversation) VALUES (?, ?, ?)''', (user_id, timestamp, conversation))
    conn.commit()
    conn.close()

    return jsonify({'status': 'success'}), 200
```

- user_id: 사용자의 고유 ID, 사용자 식별에 사용
- conversation: 사용자가 입력한 대화 내용. 이는 텍스트 형식으로 저장
- timestamp: 음성 데이터가 업로드된 시점을 기록
- conn: SQLite 데이터베이스와의 연결 객체
- cursor: 데이터베이스 작업을 수행하는 커서 객체

3. 과거 대화 데이터 요청 및 전송

```
@app.route('/request_past_conversation/<user_id>', methods=['GET'])
def request_past_conversation(user_id):
    conn = sqlite3.connect('conversation_data.db')
    cursor = conn.cursor()
    cursor.execute('''SELECT conversation FROM ConversationData WHERE user_id = ?''', (user_id,))
    data = cursor.fetchall()
    conn.close()

    if data:
```

```
        return jsonify({'conversations': [row[0] for row in data]}), 200
    else:
        return jsonify({'error': 'No data found'}), 404
```

- data: 데이터베이스 조회 결과를 저장하는 변수. 특정 사용자 ID 에 해당하는 모든 대화 데이터를 포함한다.

4. 현재 대화 데이터 요청 및 전송.

```
@app.route('/request_current_conversation/<user_id>', methods=['GET'])
def request_current_conversation(user_id):
    conn = sqlite3.connect('conversation_data.db')
    cursor = conn.cursor()
    cursor.execute('''
        SELECT conversation FROM ConversationData WHERE user_id = ?
        ORDER BY timestamp DESC LIMIT 1
    ''', (user_id,))
    data = cursor.fetchone()
    conn.close()

    if data:
        return jsonify({'conversation': data[0]}), 200
    else:
        return jsonify({'error': 'No data found'}), 404
```

5. 데이터를 NLP 분석 실시

```
def analyze_consistency(past_conversations, current_conversation):
    vectorizer = joblib.load('vectorizer.pkl')
    model = joblib.load('consistency_model.pkl')

    all_conversations = past_conversations + [current_conversation]
    X = vectorizer.transform(all_conversations)

    # 모델을 통해 일관성 점수 계산 (예시)
    consistency_scores = model.predict_proba(X)[:, 1]
```

```
    consistency_score = consistency_scores[-1]  # 현재 대화의 일관성 점수

    return consistency_score
```

- past_conversations: 사용자의 과거 대화 데이터를 포함한 리스트. 텍스트 형식으로 되어있다.
- current_conversation: 사용자의 현재 대화 데이터. 텍스트 형식으로 되어 있다
- vectorizer: 텍스트 데이터를 벡터 형식으로 변환하는 벡터라이저 객체. 텍스트 데이터를 숫자 벡터로 변환합니다.
- model: 사전 학습된 기계 학습 모델 객체입니다.
- all_conversations: 과거 대화와 현재 대화를 합친 리스트
- X: 벡터라이저를 통해 변환된 대화 데이터의 벡터 형식
- consistency_scores: 모델을 통해 예측된 일관성 점수의 확률값
- consistency_score: 현재 대화의 일관성 점수

6. 분석된 데이터를 기반으로 일관성을 평가

```
@app.route('/analyze_conversation/<user_id>', methods=['GET'])
def analyze_conversation(user_id):
    conn = sqlite3.connect('conversation_data.db')
    cursor = conn.cursor()
    cursor.execute('''SELECT conversation FROM ConversationData WHERE user_id = ?''',
(user_id,))
    past_conversations = [row[0] for row in cursor.fetchall()]

    cursor.execute('''
        SELECT conversation FROM ConversationData WHERE user_id = ?
```

```
        ORDER BY timestamp DESC LIMIT 1
    ''', (user_id,))
    current_conversation = cursor.fetchone()

    if current_conversation:
        current_conversation = current_conversation[0]
        consistency_score = analyze_consistency(past_conversations, current_conversation)

        cursor.execute('''INSERT INTO AnalysisResults (user_id, timestamp, consistency_score)
VALUES (?, ?, ?)''', (user_id, datetime.now().strftime('%Y-%m-%d %H:%M:%S'), consistency_score))
        conn.commit()
        conn.close()

        return jsonify({'consistency_score': consistency_score}), 200
    else:
        conn.close()
        return jsonify({'error': 'No data found'}), 404
```

2.2. 요소 설계

2.2.1. **Process**

프로세스 2. 대화 데이터를 DB 에 저장:

1. 데이터베이스 연결 및 커서 생성

    - conn = sqlite3.connect('conversation_data.db')

    - cursor = conn.cursor(): 데이터베이스와의 연결을 설정하고, SQL 문을 실행할 커서를 생성, 데이터베이스 작업을 수행하기 위해서는 연결 객체와 커서 객체 필요

2. 데이터 삽입

    - cursor.execute('''INSERT INTO ConversationData (user_id, timestamp, conversation) VALUES (?, ?, ?)''', (user_id, timestamp, conversation)): SQL INSERT 문을 사용하여

ConversationData 테이블에 새로운 레코드를 삽입, user_id, timestamp, conversation 필드를 각각의 값으로 채움

프로세스 3. 과거 대화 데이터 요청 및 전송

1. 응답 반환

- if data: return jsonify({'conversations': [row[0] for row in data]}), 200: 조회된 데이터가 있는 경우, 이를 JSON 형식으로 변환하여 응답으로 반환, 결과는 리스트로 감싸서 JSON 객체로 반환

프로세스 4. 현재 대화 데이터 요청 및 전송.

1. 프로세스 3 과 같은 프로세스를 가진다.

프로세스 5. 데이터를 NLP 분석 실시:

1. 텍스트 데이터 벡터화:

vectorizer.transform(all_conversations): 텍스트 데이터를 숫자 벡터로 변환. TF-IDF 벡터라이저를 사용하여 텍스트 데이터를 벡터로 변환.

2. 기계 학습 모델을 통한 일관성 점수 예측:

model.predict_proba(X)[:, 1]: 벡터화 된 텍스트 데이터를 사용하여 각 대화에 대한 일관성 점수를 확률 값으로 예측.

3. 현재 대화의 일관성 점수 추출:

consistency_scores[-1]: 모든 대화에 대한 일관성 여부를 예측한 후, 가장 마지막 대화(즉, 현재 대화)에 대한 일관성 여부를 추출.

프로세스 6. 분석된 데이터를 기반으로 일관성을 평가:

1. 일관성 여부 판단 및 저장:

   consistency_score = analyze_consistency(past_conversations, current_conversation[0]): 현재 대화 데이터가 존재하는 경우, 과거 대화 데이터와 현재 대화 데이터를 사용하여 일관성 여부를 판단.

2.2.2. **Data**

이름	**타입**	**범위**	**출력 프로세스**	**입력 프로세스**	**비고**
user_id	문자열	사용자의 고유 ID	대화 데이터 저장, 대화 데이터 조회, 일관성 분석 결과 반환	대화 데이터 저장, 과거 대화 데이터 요청, 현재 대화 데이터 요청, 일관성 분석 요청	사용자를 식별하는 고유 ID
conversation	문자열	대화 텍스트 데이터	대화 데이터 조회, 일관성 분석 결과 반환	대화 데이터 저장, 과거 대화 데이터 요청, 현재 대화 데이터 요청, 일관성 분석 요청	사용자의 대화 내용
timestamp	문자열	yyyy-mm-dd hh:mm:ss	대화 데이터 저장, 일관성 분석 결과 반환	대화 데이터 저장, 현재 대화 데이터 요청, 일관성 분석 요청	대화 데이터 저장 시간
past_ conversations	리스트	대화 텍스트 데이터 목록	일관성 분석	과거 대화 데이터 요청, 일관성 분석 요청	사용자의 과거 대화 데이터
current_ conversation	문자열	대화 텍스트 데이터	일관성 분석 결과 반환	현재 대화 데이터 요청, 일관성 분석 요청	사용자의 현재 대화 데이터
consistency	불리언	True,	일관성 분석 결과 반환	일관성 분석 요청	사용자의

		False			대화 일관성 duqn
response	JSON 객체	-	대화 데이터 저장 응답, 대화 데이터 조회 응답, 일관성 분석 결과 응답	대화 데이터 저장, 대화 데이터 조회, 일관성 분석 요청	클라이언트로 반환되는 응답 데이터

2.2.3. **저장소**

저장소 이름	목적	사용 프로세스	구성 요소
사용자 정보 데이터베이스 (UserDB)	사용자 개인 정보 및 인증 데이터 저장	사용자 정보 조회, 업데이트	Users 테이블
대화 데이터베이스 (ConversationDB)	사용자의 대화 데이터 저장	대화 데이터 저장, 과거 대화 데이터 요청, 현재 대화 데이터 요청, 일관성 분석 요청	Conversation 테이블
일관성 분석 결과 데이터베이스 (AnalysisResultsDB)	대화 데이터 분석 결과 저장	일관성 분석 결과 저장 및 조회	Result 테이블

1. 사용자 정보 데이터베이스 (UserDB)

목적: 사용자 개인 정보 및 인증 데이터 저장

구성 요소: Users 테이블:

- user_id (문자열, PRIMARY KEY): 사용자 고유 ID
- username (문자열): 사용자 이름
- password (문자열): 사용자 암호
- email (문자열): 사용자 이메일

2. 대화 데이터베이스 (ConversationDB)

목적: 사용자의 대화 데이터를 저장

구성요소: Conversation 테이블:

- id (정수, PRIMARY KEY AUTOINCREMENT): 대화 데이터의 고유 ID
- user_id (문자열): 대화 데이터를 제공한 사용자의 ID
- timestamp (문자열): 대화 데이터가 저장된 시간
- conversation (문자열): 대화 내용

3. 일관성 분석 결과 데이터베이스 **(**AnalysisResultsDB**)**

목적: 대화 데이터의 분석 결과를 저장하여, 사용자의 대화 일관성 여부를 관리

구성 요소: Result 테이블

- id (정수, PRIMARY KEY AUTOINCREMENT): 분석 결과의 고유 ID
- user_id (문자열): 분석 결과가 해당하는 사용자의 ID
- timestamp (문자열): 분석 결과가 저장된 시간
- consistency (불리언): 대화 일관성 여부

3. 기타사항

FJ_002를 통해 사용자와의 답변을 바탕으로 일관성 여부를 수집하고 이 결과를 바탕으로 CP_003의 발화 생성 알고리즘을 사용하여 사용자와 대화를 지속한다.