프로젝트명 : 실시간 한영 수어 번역을 위한 어플리케이션 개발

캡스톤 디자인II, 중간보고서

Version 1.0

개발 팀원 명(팀리더): 이경림
송채린
김하현

대표 연락처: 010-2339-4916
e-mail: kr031106@naver.com

캡스톤 디자인 II 중간보고서 내용

1. 요구사항 정의서에 명세된 기능에 대하여 현재까지 진척된 결과 및 그 내용을 기술하시오.

(1) 실행파일 및 환경설치 문서

- 실행파일 : ‘realtime_predict.py‘
  · Mediapipe 라이브러리를 활용하여 실시간으로 손 관절 포인트를 추출한다.
  · 추출된 좌표 데이터는 학습된 ‘gesture_lstm_model_dual.h5‘를 통해 단어 단위 예측을 수행한다.
  · 이후 ‘my_finetuned_t5_model‘을 불러와 단어열을 자연어 문장으로 변환하여 출력한다.
  · 이 과정을 통해 카메라 입력 → 관절 인식 → 단어 예측 → 문장 생성까지 일련의 파이프라인이 구현된 상태이다.

- 환경설치 문서 : ‘requirements.txt‘
  · 프로젝트 실행에 필요한 필수 라이브러리와 버전을 명시하였다.
  · 해당 파일을 사용하면 모든 팀원이 동일한 개발 환경을 빠르게 세팅할 수 있다.
  · VS Code에서 requirements.txt 사용하는 방법:
    1. VS Code에서 프로젝트 폴더를 연다.
    2. 터미널(단축키: ‘Ctrl+‘‘)을 열고 Python 가상환경을 생성 및 활성화한다.

```
python -m venv .venv
source .venv/bin/activate   # Mac/Linux
.venv\Scripts\activate      # Windows
```

    3. requirements.txt에 명시된 라이브러리를 설치한다.

```
pip install -r requirements.txt
```

    4. VS Code 하단의 Python 인터프리터를 .venv로 변경하여 동일한 환경에서 실행할 수 있도록 설정한다.

- 소스코드 (realtime_predict.py)

```
import cv2
import mediapipe as mp
import numpy as np
from keras.models import load_model
import joblib
from collections import deque
from PIL import ImageFont, ImageDraw, Image
import threading
import torch # 💡 T5 모델 사용을 위해 torch 라이브러리 추가
from transformers import T5ForConditionalGeneration, T5TokenizerFast as T5Tokenizer # 💡 T5 모델 클래스 추가

# === 전역 변수 ===
sentence_words = []
prediction_result = ("", 0.0)
generated_sentence = "" # 💡 생성된 최종 문장을 저장할 변수

MODEL_PATH = "models/gesture_lstm_model_dual.h5"
ENCODER_PATH = "processed_lstm/label_encoder_lstm_dual.pkl"
FRAMES_PER_SAMPLE = 15
FONT_PATH = "C:/Windows/Fonts/malgun.ttf" # 💡 폰트 경로는 사용자 환경에 맞게 설정해주세요.

CONFIDENCE_THRESHOLD = 0.85

# 💡 1. T5 모델과 토크나이저 로드 (프로그램 시작 시 한 번만 실행)
print("▶ T5 모델을 로드하는 중입니다...")
# "KETI-AIR/ke-t5-small"은 한국어에 특화된 경량 T5 모델입니다.
T5_MODEL_NAME = "./my_finetuned_t5_model"
tokenizer = T5Tokenizer.from_pretrained(T5_MODEL_NAME)
t5_model = T5ForConditionalGeneration.from_pretrained(T5_MODEL_NAME)
print("✅ T5 모델 로드 완료!")

# 💡 2. T5 모델을 사용하여 문장을 생성하는 함수 (핵심 변경 부분)
def generate_sentence_with_t5(words):
    """T5 모델을 사용해 단어 목록을 자연스러운 문장으로 변환"""
    global generated_sentence

    if not words:
        return

    # 1. ['나', '집', '가다'] 형태의 리스트를 "나,집,가다" 형태의 문자열로 변환합니다.
    #    (리스트 안의 원소가 numpy.str_ 이어도 문제없이 처리됩니다.)
    word_sequence = ", ".join(words)

    # 2. 훈련 데이터(train_dataset.json)와 '완벽하게' 동일한 형식의 프롬프트를 생성합니다.
    prompt = f"문장 생성: {word_sequence}"
    print(f"📝 T5 입력: '{prompt}'") # 디버깅용으로 실제 입력값 확인

    # 3. 문장을 토큰(숫자)으로 변환
    inputs = tokenizer(prompt, return_tensors="pt")
```

```
    # 4. 모델을 통해 문장 생성
    with torch.no_grad():
        outputs = t5_model.generate(
            inputs.input_ids,
            max_length=64,
            num_beams=5,
            early_stopping=True,
            repetition_penalty=2.0,
            no_repeat_ngram_size=2
        )

    # 5. 생성된 토큰을 다시 텍스트로 변환
    result_sentence = tokenizer.decode(outputs[0], skip_special_tokens=True)
    generated_sentence = result_sentence

    print(f"✅ T5 생성 문장: {generated_sentence}")


# === 한글 텍스트 출력 ===
def draw_korean_text(img, text, position, font_size=32, color=(0, 255, 0)):
    font = ImageFont.truetype(FONT_PATH, font_size)
    img_pil = Image.fromarray(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
    draw = ImageDraw.Draw(img_pil)
    draw.text(position, text, font=font, fill=color)
    return cv2.cvtColor(np.array(img_pil), cv2.COLOR_RGB2BGR)

# === 예측 스레드 ===
def predict_in_thread(sequence):
    global sentence_words, prediction_result, generated_sentence

    prediction = model.predict(sequence, verbose=0)
    predicted_label = label_encoder.inverse_transform([np.argmax(prediction)])[0]
    confidence = np.max(prediction)

    prediction_result = (predicted_label, confidence)

    if confidence < CONFIDENCE_THRESHOLD:
        return

    # 마지막 단어와 다를 경우에만 단어를 처리
    if not sentence_words or sentence_words[-1] != predicted_label:
        if predicted_label == "OK": # 'OK' 제스처가 인식되면
            if sentence_words:
                # 💡 3. T5 문장 생성 함수를 별도 스레드에서 호출
                threading.Thread(target=generate_sentence_with_t5, args=(list(sentence_words),), daemon=Tru
e).start()
                sentence_words.clear() # 단어 목록 초기화
        else:
            generated_sentence = "" # 새로운 단어가 추가되면 이전 문장 결과는 지움
            sentence_words.append(predicted_label)
            print(f"➕ 단어 추가: {predicted_label} (현재 리스트: {sentence_words})")


# === 모델 및 리소스 로드 ===
model = load_model(MODEL_PATH)
label_encoder = joblib.load(ENCODER_PATH)

mp_hands = mp.solutions.hands
hands = mp_hands.Hands(static_image_mode=False, max_num_hands=2)
mp_drawing = mp.solutions.drawing_utils

frame_buffer = deque(maxlen=FRAMES_PER_SAMPLE)
```

```
cap = cv2.VideoCapture(0)
print("▶ 양손 실시간 예측 시작 - 'q' 누르면 종료")

frame_count = 0

# === 메인 루프 ===
while cap.isOpened():
    ret, frame = cap.read()
    if not ret:
        break

    frame = cv2.flip(frame, 1)
    rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    result = hands.process(rgb)

    # ... (기존 손 인식 및 좌표 추출 코드는 동일) ...
    hands_data = {}
    if result.multi_hand_landmarks and result.multi_handedness:
        for hand_landmarks, hand_handedness in zip(result.multi_hand_landmarks, result.multi_handedness):
            hand_type = hand_handedness.classification[0].label
            one_hand = [lm for lm in hand_landmarks.landmark]
            hands_data[hand_type] = (one_hand, hand_landmarks)
            mp_drawing.draw_landmarks(frame, hand_landmarks, mp_hands.HAND_CONNECTIONS)

    left_coords = [0.0]*63
    if 'Left' in hands_data:
        left_coords = [coord for lm in hands_data['Left'][0] for coord in (lm.x, lm.y, lm.z)]

    right_coords = [0.0]*63
    if 'Right' in hands_data:
        right_coords = [coord for lm in hands_data['Right'][0] for coord in (lm.x, lm.y, lm.z)]

    one_frame = left_coords + right_coords

    if len(one_frame) == 126:
        frame_buffer.append(one_frame)

    frame_count += 1
    if len(frame_buffer) == FRAMES_PER_SAMPLE and frame_count % 3 == 0:
        sequence = np.array(frame_buffer).reshape(1, FRAMES_PER_SAMPLE, 126)
        threading.Thread(target=predict_in_thread, args=(sequence,), daemon=True).start()

    # --- 화면 출력 부분 ---
    # 1. 현재 인식 중인 단어 표시
    label, conf = prediction_result
    if label:
        text = f"{label} ({conf:.2f})"
        frame = draw_korean_text(frame, text, (10, 30), font_size=32, color=(255, 255, 0)) # 노란색

    # 2. 현재까지 입력된 단어 목록 표시
    words_text = " ".join(sentence_words)
    frame = draw_korean_text(frame, words_text, (10, 80), font_size=40, color=(255, 255, 255)) # 흰색
```

```
    # 💡 4. T5가 생성한 최종 문장 표시
    if generated_sentence:
        frame = draw_korean_text(frame, generated_sentence, (10, 130), font_size=40, color=(0, 255, 0)) # 초록색

    cv2.imshow("Sign Language Sentence Generator (T5)", frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()
```

(2) 모델 학습 및 개선 과정

- LSTM 모델 학습 :
  · ‘collect_hand_landmark_samples‘ 함수를 통해 손 관절 포인트 데이터를 수집하고, ‘preprocess_dual_hand_csv_lstm‘로 전처리하여 시퀀스 데이터를 구성하였다.
  · ‘train_lstm_model_dual‘을 통해 양손 입력(30프레임 × 126 관절 포인트)을 학습, 단어 단위 예측 성능을 확보하였다.
  · 학습 시 과적합 문제가 발생했으나, 검증 손실을 함께 모니터링하고 Early Stopping을 적용하여 일반화 성능을 확보하였다.

- T5 모델 파인튜닝 :
  · 초기에는 다국어 모델을 사용하여 한국어 문장이 아닌 출력이 반복되었으나, 한국어 특화 T5 모델로 교체 후 파인튜닝을 진행하여 자연스러운 한국어 문장을 안정적으로 생성하였다.
  · ‘train_t5_model‘ 학습 루프에서 조기 종료 기법을 적용하여 성능 개선 없는 구간을 자동으로 중단, 최적의 모델을 저장하도록 개선하였다.

- 실시간 문장 생성 :
  · ‘generate_sentence_with_t5‘ 함수를 통해 입력된 단어열을 “문장 생성: …” 형식의 프롬프트로 변환하고, T5 모델이 자연스러운 문장을 생성하도록 구현하였다.
  · 반복 억제(‘no_repeat_ngram_size‘, ‘repetition_penalty‘) 등을 적용하여 출력 품질을 높였다.

(3) 서버 구축 및 통신 인프라

- 당초에는 온디바이스 실행을 목표로 하였으나, 모델 크기 문제를 고려하여 **도커 기반 서버-클라이언트 구조**로 전환하는 방안을 마련하였다.

- Flask/FastAPI 기반 API 서버와 엔드포인트(‘POST /infer‘, ‘WS /stream‘) 설계를 진행하였으며, 이를 통해 실시간 추론 결과를 반환하는 구조를 구상하였다.

- 다만, 현재 시점에서는 서버 전체 구축까지는 완료되지 않았으며, **카메라를 활용한 양방향 영상 통신 기능**이 정상적으로 동작함을 확인한 단계에 있다.

(4) 애플리케이션 개발

- Android 앱에서 **카메라 캡처와 영상통화 화면 출력**까지의 흐름을 구현하였다.

- UI/UX는 영상통화 화면 내에 **한국어 번역 결과**를 크게 표시하여 가독성을 높였으며, 오류 발생 시 사용자에게 안내 메시지를 제공하도록 하였다.

- 다만, 현재 시점에서는 **서버 전송 및 영어 번역 기능은 아직 구현되지 않은 상태**이며, 향후 연동 예정 단계에 있다.

- 서버 통신은 WebSocket 기반으로 구현하여 실시간성을 확보하였고, 네트워크 오류 발생 시 재시도 및 fallback 메시지를 출력하도록 설계하였다.

(5) 문제 해결 사례

- 모델 크기와 온디바이스 구동 한계를 극복하기 위해 서버 구조를 적용.

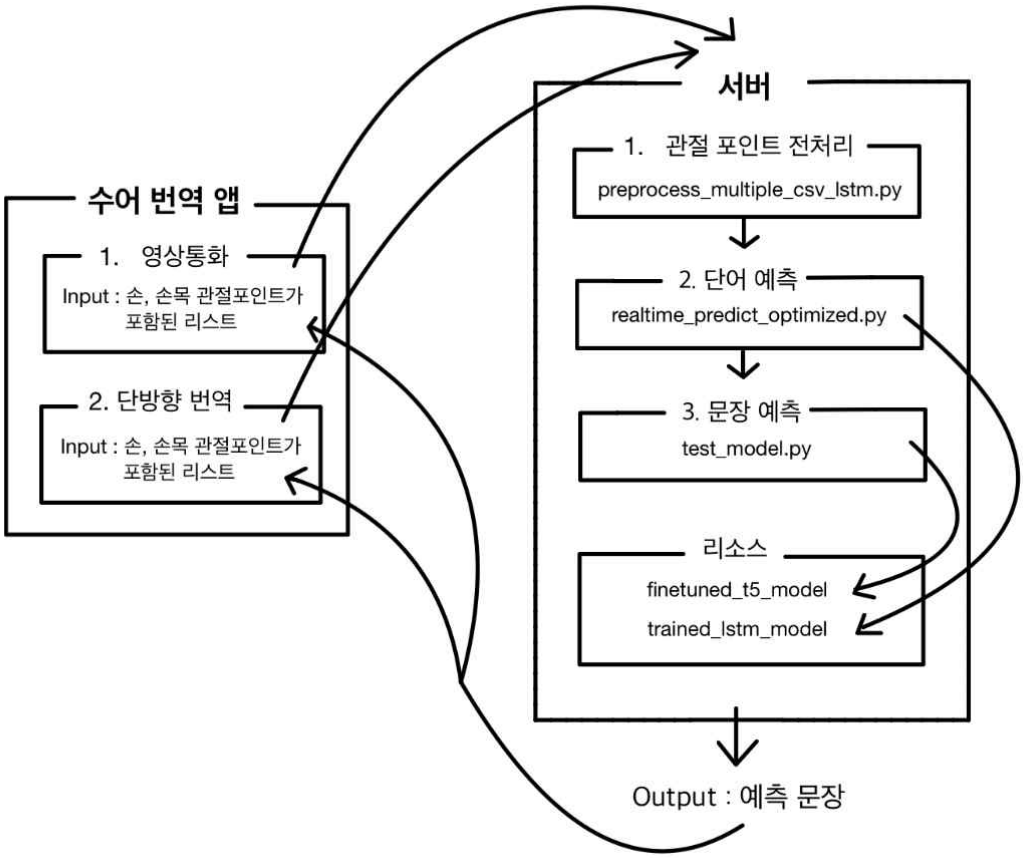
- 수어의 다의성 문제를 데이터셋 확충 및 좌표계 변환(화면 기준 → 손목 기준)으로 완화.

- 학습 시간 문제를 구글 코랩 GPU/TPU 가속기를 통해 1/5로 단축.

- 과적합 문제를 검증 기반 해석 및 Early Stopping으로 개선.

- 실시간 통신 구조의 확장성 문제를 Mesh 구조에서 **SFU(LiveKit)** 구조로 전환하여 대역폭 낭비를 최소화.

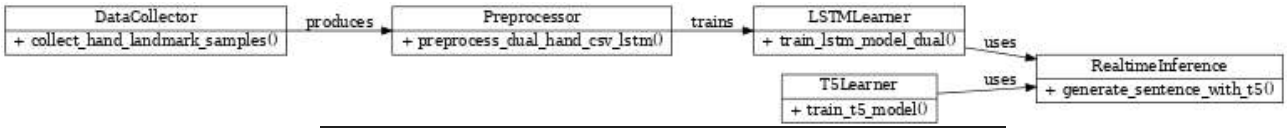
- 시스템(하드웨어, 시스템)구성도, 또는 소프트웨어 아키텍처



- 기능별 상세 요구사항(또는 유스케이스)

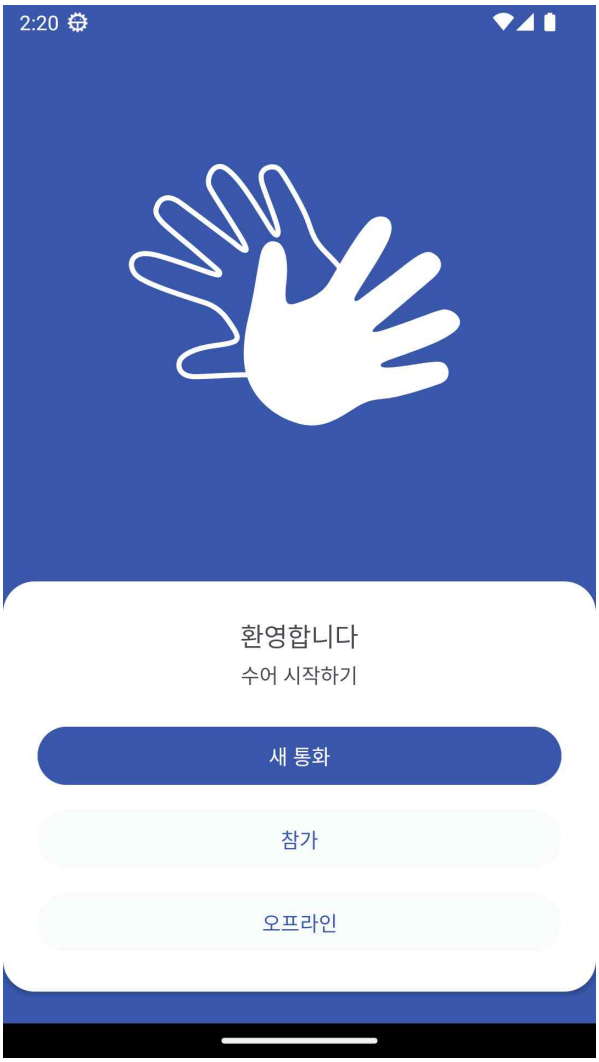
기능명	상세 요구사항	구현 결과
손 관절 포인트 인식	사용자가 카메라(또는 영상통화 화면)에 손을 움직이면, realtime_predict.py에서 Mediapipe 라이브러리를 통해 양손/손목 관절 좌표를 실시간 추출	☑ 구현 완료 및 정상 동작
단어 예측	추출된 관절 포인트 시퀀스를 gesture_lstm_model_dual.h5 모델에 입력하여, 사전에 학습된 한국수어 단어 단위로 예측	☑ 구현 완료 및 정상 동작
문장 생성	예측된 단어들을 입력값으로 하여, my_finetuned_t5_model을 불러와 자연어 처리 및 문장 구성	☑ 구현 완료 및 정상 동작
서버 구축	Dockerfile 작성(모델/코드 분리 마운트), env로 Papago 키·모델 경로 주입, 이미지 빌드/실행 문서화	✕ 미구현
번역 API 연동	구성된 한국어 문장을 외부 번역 API를 통해 영어 문장으로 변환	✕ 미구현
어플리케이션 개발	사용자 친화적인 UI/UX 화면을 통해 번역 결과를 실시간 표시	⚠ 구현 중

- 설계 모델(클래스 다이어그램, 클래스 및 모듈 명세서)




```
classDiagram
    class DataCollector {
        + collect_hand_landmark_samples()
    }

    class Preprocessor {
        + preprocess_dual_hand_csv_lstm()
    }

    class LSTMLearner {
        + train_lstm_model_dual()
    }

    class T5Learner {
        + train_t5_model()
    }

    class RealtimeInference {
        + generate_sentence_with_t5()
    }

    DataCollector --> Preprocessor : produces
    Preprocessor --> LSTMLearner : trains
    LSTMLearner --> RealtimeInference : uses
    T5Learner --> RealtimeInference : uses
```

- UI 분석/설계 모델/프로토타입



2. 프로젝트 수행을 위해 적용된 추진전략, 수행 방법의 결과를 작성하고, 만일 적용과정에서 문제점이 도출되었다면 그 문제를 분석하고 해결방안을 기술하시오.

- 문제해결을 위해 적용한 방법(또는 기법) 결과, (문제점, 해결방안)

실시간 수어 번역 애플리케이션을 개발하기 위해, 데이터 수집·모델 학습·서버 인프라·애플리케이션 모듈 구현을 병렬적으로 추진하는 전략을 적용하였다. 추진 과정에서 다음과 같은 문제점과 해결 방안을 도출하였다.

1. 학습 모델의 크기 및 온디바이스 구동 한계
- 온디바이스 실행 대신 **도커 기반 서버-클라이언트 구조**를 설계하여, 학습 모델을 서버에 배치하고 모바일 앱은 REST/WS 통신으로 결과를 전달받는 방식으로 전환하였다.

2. 수어 데이터의 모호성과 다의성
- 단순히 학습 데이터에 의존하기보다, **데이터셋을 추가 확보·확장**하고 손 관절 좌표를 화면 기준이 아닌 **손목 기준 좌표계**로 변환하여 인식률을 개선하였다.

3. 학습 시간 과다 소요
- GPU/TPU 하드웨어 가속기를 활용하여 학습 시간을 기존 대비 약 1/5로 단축하였다.

4. 모델 과적합 문제
- 학습 손실과 검증 손실을 함께 모니터링하여 **일반화 성능 중심의 해석**을 적용하였고, 데이터 증강·Early Stopping 등을 통해 실제 문장 예측 성능을 향상시켰다.

5. 모델 언어 불일치
- 초기에는 다국어 기반 T5 모델을 사용해 한국어가 아닌 언어가 출력되는 문제가 있었으나, 한국어 특화 모델로 교체 후 파인튜닝하여 **자연스러운 한국어 문장 생성** 결과를 얻었다.

6. 실시간 통신 구조의 확장성 부족
- WebRTC P2P 구조에서 Mesh 연결로 인해 대역폭 소모가 과다하게 발생했으나, **SFU(Selective Forwarding Unit) 기반 아키텍처**로 전환하고 LiveKit을 도입함으로써 안정적이고 확장 가능한 실시간 영상 통신 환경을 구축하였다.

이를 통해 프로젝트는 기술적 문제를 단계적으로 극복하며 **데이터 처리 효율, 모델 인식 성능, 실시간 통신 안정성**을 모두 확보할 수 있었다.

- 팀원의 책임 및 역할 수행에 대한 결과, (문제점, 해결방안)

이경림 (팀장, AI 총괄)

팀장으로서 프로젝트 전체를 총괄하며 실현 가능성이 있는 기술적 범위와 그렇지 않은 부분을 구분하는 역할을 수행하였다. 프로젝트 진행 과정에서 팀원들에게 막히는 부분이 생길 때마다 해결책을 제시하고 함께 문제를 헤쳐 나갔다. 특히, 모델 학습 및 배포 단계에서 여러 가지 중요한 문제를 직접 분석하고 해결 방안을 마련하였다.

첫 번째로, 학습한 모델의 크기가 너무 커서 원래 목표였던 온디바이스 실행이 불가능해진 상황이 발생하였다. 이를 해결하기 위해 도커를 활용하여 모델을 서버에 배치하고, 클라이언트 애플리케이션과는 네트워크 통신을 통해 결과를 주고받는 구조를 제안하였다.

두 번째로, 수어 자체의 특성으로 인해 단순 학습으로는 표현하기 어려운 단어가 존재하거나 한 가지 수어 동작이 여러 의미를 가지는 문제점에 직면하였다. 이에 대해 추가적인 데이터셋을 확보하고 학습 범위를 확대함으로써 문제를 완화하고 모델의 일반화 성능을 향상시켰다.

세 번째로, 모델 학습 과정에서 소요되는 시간이 지나치게 길어 개발 효율성을 저해하는 문제가 발생하였다. 이를 개선하기 위해 하드웨어 가속기(GPU/TPU)를 적극적으로 활용하여 학습 속도를 기존 대비 약 1/5 수준으로 줄였고, 결과적으로 학습 환경을 크게 개선할 수 있었다.

네 번째로, 학습 데이터에 과도하게 적합하는 과적합 문제가 발생하였다. 이 문제는 단순히 높은 학습 성능이 아닌 검증 성능과의 균형을 중시하는 방식으로 해석하고, 학습 곡선을 면밀히 분석하여 학습을 조정하였다. 이를 통해 모델의 일반화 성능을 개선하고 실제 문장 예측 정확도를 높일 수 있었다.

송채린 (AI 개발 담당)

문장 생성을 위한 모델 탐색과 학습을 전담하였다.

프로젝트 과정에서 문장 생성 모델을 학습시키는 중 한국어 대신 다른 언어가 출력되는 문제가 반복적으로 발생하였는데, 이를 분석한 결과 기존에 사용하던 T5 모델이 다국어에 특화된 범용 모델이었음을 발견하였다. 이에 따라 한국어에 최적화된 언어 모델로 교체하고 파인튜닝을 진행한 결과, 자연스러운 한국어 문장을 안정적으로 생성할 수 있게 되었다.

또한, 단어 데이터가 누적됨에 따라 유사한 손동작을 구분하지 못하는 문제가 드러났다. 원인을 분석한 결과, 기존 방식이 화면 좌표계를 기준으로 손 위치를 기록하는 구조

였기 때문에 같은 손동작이라도 화면 내 위치가 다르면 서로 다른 동작으로 인식되는 문제가 있었다. 이를 해결하기 위해 손목을 기준으로 좌표계를 변환하여 손 관절 좌표를 계산하는 방법을 적용하였고, 그 결과 인식률이 유의미하게 향상되었다.

더 나아가 LSTM 모델 학습 과정에서도 전략적 개선을 이루었다. 초기에는 새로운 모델을 계속 불러와 새로 학습시키는 방식을 사용했으나, 인식률이 낮은 특정 손동작에 대해 효율적인 학습이 이루어지지 않았다. 이를 해결하기 위해 기존에 학습한 모델을 저장한 뒤 다시 불러와 추가 데이터를 지속적으로 학습시키는 **점진적 학습 방식**으로 전환하였다. 이로써 데이터 효율성이 향상되었고, 특히 인식률이 낮았던 손동작들에 대해 성능을 보완할 수 있었다.

김하현 (앱 개발 담당)

실시간 영상 통신 모듈의 설계 및 구현을 담당하였다. 초기 단계에서는 WebRTC를 활용하여 Firebase Realtime Database를 시그널링 서버로 이용하는 **P2P 기반 구조**를 설계하였다. ICE 후보 교환과 SDP 세션 관리를 통해 양방향 화상 통신을 성공적으로 구현하였으며, 이를 통해 기본적인 실시간 영상 통화 기능을 확보할 수 있었다. 그러나 프로젝트가 확장되어 수어 인식 딥러닝 서버를 연결하는 과정에서 P2P Mesh 구조의 한계가 명확히 드러났다. 참여자가 늘어날수록 연결 수가 기하급수적으로 증가하여 대역폭 소모가 급격히 커졌고, 이로 인해 통신 지연이 발생하며 실시간성이 저하되었다. 이는 실시간성을 최우선으로 요구하는 수어 번역 애플리케이션의 특성과 충돌하는 치명적인 문제였다. 이를 해결하기 위해 기존 구조를 **SFU(Selective Forwarding Unit) 기반 아키텍처**로 전환하였다. 구체적으로는 LiveKit을 도입하여 서버가 각 참여자의 영상 스트림을 받아 필요한 참가자에게만 중계하도록 하였고, 이 방식으로 불필요한 대역폭 낭비를 제거하고 네트워크 효율을 크게 향상시켰다. 그 결과 다자간 통신 환경에서도 안정적이고 확장 가능한 구조를 확보할 수 있었으며, 이후 수어 인식 모델의 번역 기능을 실제로 통합·검증할 수 있는 기반을 마련하였다.

- 프로젝트 일정계획에 맞추지 못한 경우의 문제점, 해결 방안

- 문제점

프로젝트 일정상 서버를 구축하고 모델을 배포하여 앱과 실시간으로 연동하는 단계가 포함되어 있으나, 만약 해당 작업이 일정에 맞추어 완료되지 못할 경우 전체 기능 검증이 지연될 수 있다. 특히 도커 기반 환경 설정과 API 엔드포인트 구현 과정에서 예상치 못한 오류가 발생하거나 통신 구조를 안정화하는 데 시간이 더 소요된다면, 실시간 추론 결과를 앱으로 전달하는 기능이 최종 시연 단계에서 완전하지 않을 가능성이 존재한다.

- 해결 방안

이러한 상황에 대비하여 서버 구축 일정이 지연되더라도 프로젝트 전체 흐름이 중단되지 않도록 우선순위를 조정한다. 먼저 수어 인식 및 문장 예측 모델을 온전히 학습·검증해 **서버에 탑재 가능한 상태**로 준비하고, 애플리케이션 측에서는 카메라 캡처와 번역 결과 표시 UI를 선행 구현하여 서버와의 통신이 연결되는 즉시 통합할 수 있도록 한다. 또한, 서버 구축은 모놀리식 구조가 아닌 도커 기반의 모듈화된 단계별 구현으로 진행하여, 부분적으로라도 기능 검증이 가능하도록 한다. 이를 통해 일정 차질이 발생하더라도 프로젝트의 핵심 기능은 유지하고, 이후 남은 시간을 활용해 서버 연동을 점진적으로 완성해 나갈 수 있도록 한다.

[별첨1]

프로젝트명 : 실시간 한영 수어 번역을 위한 어플리케이션 개발

소프트웨어 요구사항 정의서

Version 1.0

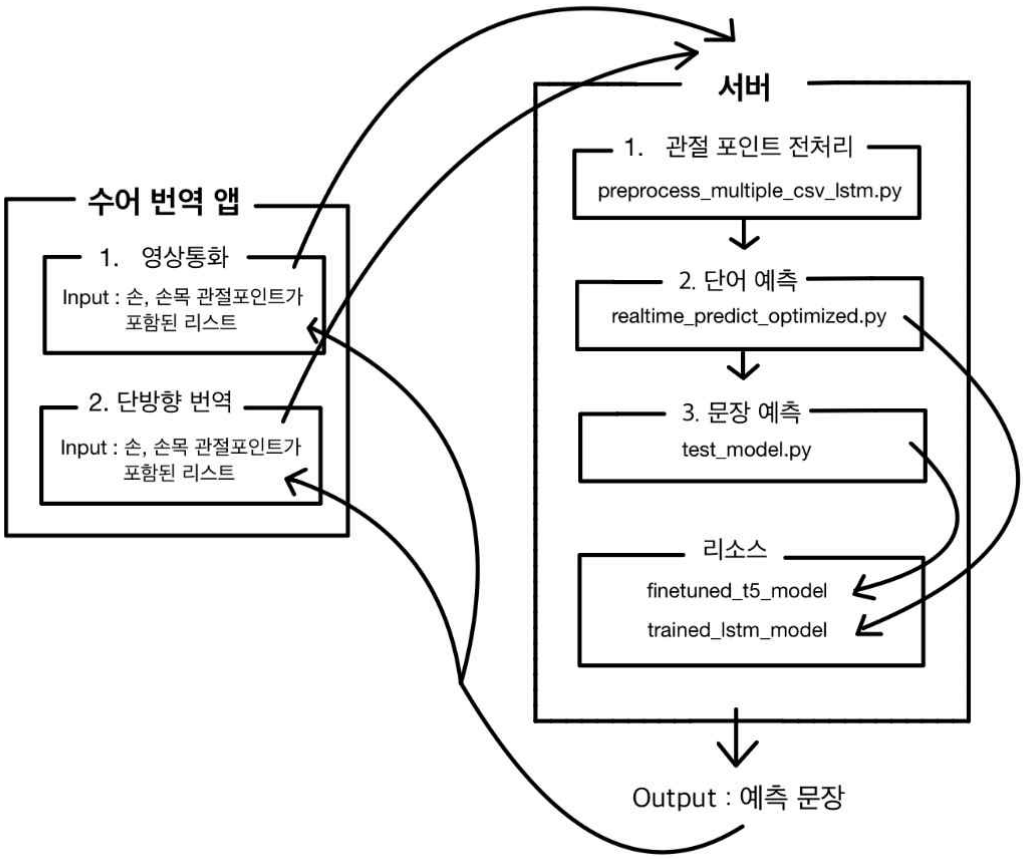
개발 팀원 명(팀리더): 이경림
송채린
김하현

대표 연락처: 010-2339-4916
e-mail: kr031106@naver.com

목차

1. 시스템 개요

- **사업 목표 :** 청각장애인을 위한 실시간 한영 수어 번역 어플리케이션 개발
- **추진 범위**
  - 핵심 기능
    · 영상통화 중 수어 → 텍스트 실시간 번역
    · 스마트폰 카메라 기반 외국어 수어 실시간 번역
  - **개발 계획**
    · 도커 기반 서버 환경 구축
    · 사용자 친화적 수어 번역 모바일 앱 구현



2. 시스템 장비 구성요구사항

요구사항 고유번호		ECR-001		
요구사항 명칭		스마트폰/태블릿(1080p)		
요구사항 분류		시스템 장비 구성	응락수준	필수
요구사항 상세설명	정의	수어 입력 촬영 및 번역 결과 표시용 사용자 단말		
	세부내용	· 규격: 고해상도 카메라(최소 1080p 이상) · 수량: 사용자 수에 따라 유동적 · 용도: 수어 입력 영상 촬영 및 디스플레이		

요구사항 고유번호		ECR-002		
요구사항 명칭		GPU 서버(학습/추론)		
요구사항 분류		시스템 장비 구성	응락수준	필수
요구사항 상세설명	정의	수어 인식(LSTM)·문장 생성(T5) 모델 학습/고속 추론		
	세부내용	· 규격: NVIDIA RTX 3090 이상 또는 A100급 GPU · 수량: 모델 학습 시 1대 이상 · 용도: 수어 인식 및 자연어 처리 모델 학습 및 추론		

요구사항 고유번호		ECR-003		
요구사항 명칭		개발환경(언어/라이브러리)		
요구사항 분류		시스템 장비 구성	응락수준	필수
요구사항 상세설명	정의	모델·앱 개발/빌드 도구 및 런타임		
	세부내용	· Python 3.10+ , · TensorFlow/Keras, PyTorch, · HuggingFace Transformers, · OpenCV · MediaPipe · Android Studio(Gradle, SDK 26+)		

요구사항 고유번호		ECR-004		
요구사항 명칭		모바일 데이터 또는 Wi-Fi 환경		
요구사항 분류		시스템 장비 구성	응락수준	필수
요구사항 상세설명	정의	번역/통역사 연동을 위한 통신 인프라		
	세부내용	· 용도: 실시간 번역 전송 및 전문 통역사 연결 · 요구사항: 안정적인 업/다운로드 속도 보장 (예: 최소 10Mbps 이상)		

요구사항 고유번호		ECR-005		
요구사항 명칭		IDE/CI 환경		
요구사항 분류		시스템 장비 구성	응락수준	선택
요구사항 상세설명	정의	빌드·테스트 자동화 및 릴리스 파이프라인		
	세부내용	· GitHub Actions(유닛/통합/린트) · Play Store 내 배포 트랙		

3. 기능 요구사항

요구사항 고유번호		SFR-001		
요구사항 명칭		영상통화 실시간 번역		
요구사항 분류		기능	응락수준	필수
요구사항 상세설명	정의	양방향 영상통화 중 수어→텍스트 실시간 변환		
	세부내용	(1) 프레임 캡처 (2) MediaPipe Hands(양손, 21×3D) (3) LSTM 단어 예측 (4) 단어열→T5 입력 프롬프트 변환 (5) 문장 생성 (6) 화면 자막 표시		

요구사항 고유번호		SFR-002		
요구사항 명칭		카메라 모드 번역(단방향)		
요구사항 분류		기능	응락수준	필수
요구사항 상세설명	정의	단말 카메라 입력 영상의 수어를 자막화		
	세부내용	방송/강연/뉴스 등 외부 영상 인식, 화면 하단 자막		

요구사항 고유번호		SFR-003		
요구사항 명칭		문장 생성(T5 파인튜닝)		
요구사항 분류		기능	응락수준	필수
요구사항 상세설명	정의	인식 단어열을 자연스러운 한국어 문장으로 생성		
	세부내용	pko-T5-base 파인튜닝 모델(my_finetuned_t5_model) 사용, 입력 포맷 "문장 생성: 단어, 단어…"		

요구사항 고유번호		SFR-004		
요구사항 명칭		사용자 맞춤 수어 대치 사전		
요구사항 분류		기능	응락수준	선택
요구사항 상세설명	정의	신조어/지역 수어 사용자 정의 및 자동 대치		
	세부내용	CRUD UI, 사용자 스코프 저장, 대치 우선순위		

요구사항 고유번호		SFR-005		
요구사항 명칭		전문 통역사 연결(폴백)		
요구사항 분류		기능	응락수준	선택
요구사항 상세설명	정의	저신뢰/민감 대화 시 통역사 화상 연결		
	세부내용	신뢰도 임계치 미만 시 안내/선택 연결, 녹화·저장 정책 고지		

4. 성능 요구사항

요구사항 고유번호		PER-001		
요구사항 명칭		실시간성		
요구사항 분류		성능	응락수준	필수
요구사항 상세설명	정의	입력→자막 평균 1.5s 이내(최대 3s)		
	세부내용	지연 구성요소 · 캡처/전처리(≤200ms) · LSTM 추론(≤50ms) · T5 생성(≤300ms) · 네트워크(≤200ms)		

요구사항 고유번호		PER-002		
요구사항 명칭		처리량(FPS)		
요구사항 분류		성능	응락수준	필수
요구사항 상세설명	정의	30fps 입력 기준 파이프라인 유지, 저사양 단말 20fps 보장		
	세부내용	프레임 드롭률 < 5%		

요구사항 고유번호		PER-003		
요구사항 명칭		자원 사용량(모바일)		
요구사항 분류		성능	응락수준	필수
요구사항 상세설명	정의	CPU ≤ 40%, 메모리 ≤ 1GB		
	세부내용	· 경량화(TFLite 또는 int8 양자화) 고려 · 백그라운드 옵트아웃 제공		

요구사항 고유번호		PER-004		
요구사항 명칭		모델 추론 시간(서버)		
요구사항 분류		성능	응락수준	보통
요구사항 상세설명	정의	LSTM 단어 예측 < 50ms/시퀀스, T5 문장 생성 < 300ms/문장(배치 1)		
	세부내용	· FP16·FlashAttention 고려 · 프롬프트 길이 128 토큰 제한		

요구사항 고유번호		PER-005		
요구사항 명칭		가용성		
요구사항 분류		성능	응락수준	보통
요구사항 상세설명	정의	서비스 가용성 99.0% 이상		
	세부내용	서버 이중화/헬스체크/자동 재시작		

5. 인터페이스 요구사항

5-1) 사용자 인터페이스 (UI)

요구사항 고유번호		SIR-UI-001		
요구사항 명칭		사용자 편의성		
요구사항 분류		인터페이스	응락수준	필수
요구사항 상세설명	정의	사용자가 시스템을 쉽게 조작하고 주요 기능에 빠르게 접근할 수 있어야 함.		
	세부내용	1. 메인 화면은 2×2 아이콘 기반 메뉴(채널 생성, 채널 참가, 카메라 모드, 설정)로 구성하여 직관적 사용성 보장. 2. 버튼 및 텍스트는 모바일 환경에서 터치하기 쉬운 크기(44px 이상)와 간격을 준수. 3. 저사양 단말 사용자는 경량 UI 모드로 접근 가능해야 함.		
주석		근성 가이드 준수, 모바일 친화적 UI 설계 필요.		
요구사항 출처		캡스톤디자인 중간/최종 보고서 UI 분석 파트		

요구사항 고유번호		SIR-UI-002		
요구사항 명칭		정보 접근성		
요구사항 분류		인터페이스	응락수준	필수
요구사항 상세설명	정의	사용자가 필요한 설정과 정보를 쉽게 접근할 수 있어야 함.		
	세부내용	1. 언어, 수어 대치 키워드 등 사용자 맞춤 설정 가능. 2. 카메라 모드 진입 시 별도 설명 없이 실시간 번역 자막이 화면 하단에 자동 표시. 3. 청각장애인 사용자를 위해 모든 음성 안내는 자막 병행 제공.		
주석		사용자 범주에 서비스 소외계층을 포함하여 인터페이스를 설계해야 함.		
요구사항 출처		캡스톤디자인 중간/최종 보고서 UI 분석 파트		

요구사항 고유번호		SIR-UI-003		
요구사항 명칭		작업 효율성		
요구사항 분류		인터페이스	응락수준	선택
요구사항 상세설명	정의	사용자가 최소한의 조작으로 목적 기능을 빠르게 수행할 수 있어야 함.		
	세부내용	1. QR코드 기반 채널 참가 기능을 제공하여 사용자 간 영상통화 연결을 간소화. 2. 번역 시작/중지 버튼은 중앙에 배치하여 작업 흐름이 끊기지 않도록 함. 3. 오류 발생 시 팝업·진동·음성 안내를 통해 즉시 피드백 제공.		
주석		저사양 단말기에서도 최소 조작으로 주요 기능 수행 가능해야 함.		
요구사항 출처		캡스톤디자인 중간/최종 보고서 UI 분석 파트		

요구사항 고유번호		SIR-UI-004		
요구사항 명칭		정보 유용성		
요구사항 분류		인터페이스	응락수준	선택
요구사항 상세설명	정의	번역 결과와 사용자 정의 정보를 직관적으로 제공해야 함.		
	세부내용	1. 번역 결과는 화면 하단에 실시간 자막 형태로 제공. 2. 참여 코드 제공 시, 복사 버튼 포함하여 정보 공유 편의성 확보.		

		3. 시각장애인을 위해 번역 결과를 TTS(Text-to-Speech) 기능으로 읽어 주는 옵션 제공.
주석		사용자 맞춤형 사전 기능과 연동되어 정보 유용성이 강화됨.
요구사항 출처		캡스톤디자인 중간/최종 보고서 UI 분석 파트

5-2) 시스템 인터페이스 요구사항 분석 및 도출

요구사항 고유번호		SIR-SYS001		
요구사항 명칭		수어 인식 API 연동		
요구사항 분류		시스템 인터페이스	응락수준	필수
요구사항 상세설명	정의	클라이언트 앱에서 촬영한 수어 영상을 LSTM 모델 API로 전달하고 인식 결과(단어)를 수신하는 인터페이스		
	세부내용	· 인터페이스 이름: /predict-gesture · 연계 대상 시스템: 모바일 앱(송신) ↔ 수어 인식 서버(수신) · 연계 범위 및 내용: 30프레임(좌/우 손 21좌표) 기반 입력 데이터를 전달하고, 인식된 단어 라벨과 신뢰도 반환 · 연계 방식: REST API (HTTP POST, JSON 포맷) · 송신 데이터: {frames: [...], metadata: {...}} · 인터페이스 주기: 1초 단위(30fps 기반 시퀀스 처리) · 기타 고려사항: 네트워크 지연 시 캐싱 처리, 모델 버전 관리 필요		
산출정보		예측 라벨, confidence score, 로그 기록		
요구사항 출처		realtime_predict_optimized.py, train_lstm_model.py		

요구사항 고유번호		SIR-SYS002		
요구사항 명칭		문장 생성 API 연동		
요구사항 분류		시스템 인터페이스	응락수준	필수
요구사항 상세설명	정의	LSTM 단어 인식 결과를 입력으로 받아 T5 모델을 통해 자연스러운 한국어 문장을 생성하는 인터페이스		
	세부내용	· 인터페이스 이름: /generate-sentence · 연계 대상 시스템: 모바일 앱(송신) ↔ 수어 인식 서버(수신) · 연계 범위 및 내용: "문장 생성: 단어, 단어..." 형식으로 전달 → 완성된 한국어 문장 반환 · 연계 방식: REST API (HTTP POST, JSON) · 송신 데이터: {prompt: "문장 생성: 나, 집, 가다"} · 인터페이스 주기: 인식된 문장 단위 발생 시 호출 · 기타 고려사항: 최대 토큰 수 제한(128), 생성 파라미터(temperature, top-p) 관리 필요		
산출정보		생성된 문장, 디코딩 로그		
요구사항 출처		train_t5.py, train_dataset.json		

요구사항 고유번호		SIR-SYS003		
요구사항 명칭		번역 API 연동		
요구사항 분류		시스템 인터페이스	응락수준	필수
요구사항 상세설명	정의	생성된 한국어 문장을 외부 번역 API(예: Papago)를 통해 영어로 번역하는 인터페이스		
	세부내용	· 인터페이스 이름: /translate · 연계 대상 시스템: 문장 생성 서버(송신) ↔ 외부 번역 API(수신) · 연계 범위 및 내용: 한국어 문장 → 영어 문장 변환 · 연계 방식: REST API (HTTP POST, HTTPS 암호화) · 송신 데이터: {text: "저는 집에 갑니다.", source: "ko", target: "en"} · 인터페이스 주기: 문장 생성 시 호출 · 기타 고려사항: API Key 보안 관리, 호출 실패 시 재시도 백오프 적용		
산출정보		번역된 문장, API 호출 로그		
요구사항 출처		파파고 번역 API 명세서, 프로젝트 API 설계		

요구사항 고유번호		SIR-SYS004		
요구사항 명칭		사용자가 등록한 커스텀 수어 대치 사전을 CRUD 방식으로 관리하는 인터페이스		
요구사항 분류		시스템 인터페이스	응락수준	선택
요구사항 상세설명	정의	생성된 한국어 문장을 외부 번역 API(예: Papago)를 통해 영어로 번역하는 인터페이스		
	세부내용	· 인터페이스 이름: /custom-dict · 연계 대상 시스템: 모바일 앱(송신) ↔ 데이터베이스 서버(수신) · 연계 범위 및 내용: 사용자 등록·수정·삭제 요청 반영, 번역 시 자동 적용 · 연계 방식: REST API (HTTP GET/POST/PUT/DELETE, JSON) · 송신 데이터: {user_id, sign_token, replace_text, priority} · 인터페이스 주기: 사용자 요청 시 발생 · 기타 고려사항: JWT 인증 기반 접근제어, 감사로그 저장, 개인정보 암호화 저장		
산출정보		사용자 정의 수어 데이터, 변경 이력 로그		
요구사항 출처		DB 설계 문서, 커스텀 수어 기능 설계서		

6. 데이터 요구사항

요구사항 고유번호		DAR-001		
요구사항 명칭		초기 데이터 구축		
요구사항 분류		데이터	응락수준	필수
요구사항 상세설명	정의	시스템 학습과 번역 정확도를 보장하기 위해 고품질 초기 데이터셋을 확보해야 함.		
	세부내용	1. 데이터 식별: · 외부 DB: AI Hub 공개 한국 수어(KSL) 라벨링 데이터셋 · 외부 DB: 영어 수어(ASL) 공개 데이터셋 일부 · 자체 수집: 손목 기준 상대 좌표 정규화 적용 MediaPipe Hands 기반 좌표 데이터 (126차원 × 30프레임) · 문장 데이터: train_dataset.json 기반 문장 생성 학습 말뭉치 2. 데이터 관리: · 영상 데이터는 전처리 후 CSV/NumPy 배열로 변환, 학습용·검증용 분할 관리 · 데이터 수집 로그는 프로젝트 저장소에 기록 후 백업 3. 데이터 제약: · 영어 수어 데이터는 확보 난이도가 높아 일부 부족할 수 있음. · 데이터 수집 시 표정·시선 등 비수지기호 포함이 제한적일 수 있음.		

요구사항 고유번호		DAR-002		
요구사항 명칭		처리 데이터 관리		
요구사항 분류		데이터	응락수준	필수
요구사항 상세설명	정의	실시간 번역 중 생성되는 데이터(영상, 로그, 결과 텍스트 등)는 안전하게 관리되어야 함.		
	세부내용	1. 데이터 식별: · 실시간 입력: 카메라 영상 프레임 · 처리 결과: 인식 단어 라벨, 문장 생성 결과, 번역 결과 텍스트 2. 데이터 관리: · 영상 데이터는 서버에 저장하지 않고 실시간 처리 후 즉시 폐기 · 로그와 사용자 등록 사전은 AES256 암호화 DB에 저장 · 로그 데이터는 최대 30일 보관 후 자동 삭제 3. 데이터 제약: · 개인정보 포함 데이터는 저장하지 않음(비식별화 원칙 준수) · 네트워크 지연 시 데이터 유실 가능성 있음 → 재전송 로직 필요		

요구사항 고유번호		DAR-003		
요구사항 명칭		신규 데이터베이스 구축		
요구사항 분류		데이터	응락수준	필수
요구사항 상세설명	정의	신규 시스템 서비스 운영을 위해 전용 DB 구조를 설계·구축해야 함.		
	세부내용	1. 데이터 식별: · 사용자 테이블 (ID, 이메일, 언어 설정 등) · 번역 로그 테이블 (시간, 영상 ID, 결과, 신뢰도 등) · 커스텀 수어 테이블 (수어 토큰, 대응 텍스트, 우선순위 등) 2. 데이터 관리: · 클라우드 기반 NoSQL DB와 로컬 SQLite를 혼합 운영 · 사용자 요청 시 모든 데이터 즉시 삭제 가능 · 정기 백업 및 무결성 검증 수행 3. 데이터 제약: · 사용자 수 증가 시 DB 확장성 문제 발생 가능 → 수평적 확장 구조 필요 · 커스텀 사전은 사용자 개별 공간으로 분리 저장해야 함		

7. 테스트 요구사항

요구사항 고유번호		TER-001		
요구사항 명칭		테스트 방안(총괄)		
요구사항 분류		테스트	응락수준	필수
요구사항 상세설명	정의	시스템 목표 대비 품질을 검증하기 위한 전사 테스트 전략과 범위 수립		
	세부내용	1. 테스트 유형: 단위·통합·시스템·성능 / 부하·신뢰도 / 회귀·보안 / 네트워크·사용성 / 접근성 2. 테스트 환경: 모바일(Android), 서버(GPU/CPU), 네트워크(4G/5G/Wi-Fi) 3. 완료 기준(Exit Criteria): · 주요 결함(CRITICAL/MAJOR) 0건 · 성능 KPI 충족(응답≤1.5s, FPS≥20~30) · 보안 취약점 High 0건, · 기타 고려 사항 형상관리 브랜치별(feature/dev/main) 테스트 게이트 운영, 자동화 우선 · 산출정보 테스트 계획서, 테스트 케이스/시나리오, 결함 리포트, 주간 테스트 요약		

요구사항 고유번호		TER-002		
요구사항 명칭		단위 테스트(Unit)		
요구사항 분류		테스트	응락수준	필수
요구사항 상세설명	정의	모듈 단위 기능(전처리, 모델 I/O, 포맷 변환, UI 유틸) 검증		
	세부내용	1. 전처리: CSV→numpy 변환, 시퀀스 길이(30), 결측/문자 값 처리, 라벨 인코딩 검증 2. 모델 I/O: LSTM 입력/출력 텐서 형태, T5 프롬프트("문장 생성: …")·토큰 길이 상한 검증 3. 유틸: 신뢰도 계산, 누적 단어 deque, 자막 렌더 함수 4. 커버리지: 라인 기준 ≥70%		

요구사항 고유번호		TER-003		
요구사항 명칭		통합/시스템 테스트(E2E)		
요구사항 분류		테스트	응락수준	필수
요구사항 상세설명	정의	입력(카메라)→수어 인식(LSTM)→문장 생성(T5)→번역→UI 자막 출력까지 전 흐름 검증		
	세부내용	1. 정상 시나리오: 단어열→문장→영문 번역→자막 표시 2. 예외 시나리오: 카메라 권한 거부, 프레임 드롭, 모델 불러오기 실패, 번역 API 실패(재시도/대체문구) 3. 데이터 시나리오: 신조어/사용자사전 대치, 미등록 수어 처리		

요구사항 고유번호		TER-004		
요구사항 명칭		성능/부하 테스트		
요구사항 분류		테스트	응락수준	필수
요구사항 상세설명	정의	실시간성(지연·FPS)과 처리량/자원 사용 한계 검증		
	세부내용	1. 지연 목표: · 입력→자막 평균 ≤1.5s, · 최대 ≤3s(구간별: 캡처≤200ms, LSTM≤50ms, T5≤300ms, 네트워크≤200ms) 2. FPS 목표: 권장 단말 30fps, 저사양 20fps 이상, 프레임 드롭 <5% 3. 자원: 모바일 CPU≤40%, 메모리≤1GB / 서버 추론 LSTM<50ms, T5<300ms(배치1) 4. 부하 : 동시 문장 생성 2개 이상 시 응답 안정성		

요구사항 고유번호		TER-005		
요구사항 명칭		신뢰도/회귀 테스트		
요구사항 분류		테스트	응락수준	보통
요구사항 상세설명	정의	다양한 환경에서 정확도·일관성 확인 및 버전 변경 시 품질 회귀 방지		
	세부내용	1. 환경 매트릭스: 조도(저/중/고), 배경(단색/복잡), 동작속도(느림/보통/빠름)별 F1/정확도 측정 2. 버전 회귀: 모델/앱/사전 변경 시 기준셋 재실행, 임계치 하회 시 릴리스 보류		

요구사항 고유번호		TER-006		
요구사항 명칭		보안/네트워크 테스트		
요구사항 분류		테스트	응락수준	필수
요구사항 상세설명	정의	데이터/통신/권한·인증과 외부 API 연계 보안 검증		
	세부내용	1. 통신: HTTPS/TLS1.2+, 중간자·재전송 방지, 키 고정(가능 시) 2. 인증/권한: JWT 만료/재발급, 권한 우회 차단, 관리자 기능 접근 제한 3. 데이터: 영상 미저장 또는 즉시 폐기, 로그/사전 AES256 저장, 삭제 요청 즉시 삭제 4. API: 번역 키 비노출, 속도 제한/백오프, 실패 시 폴백 메시지		

요구사항 고유번호		TER-007		
요구사항 명칭		사용성/접근성 테스트		
요구사항 분류		테스트	응락수준	보통
요구사항 상세설명	정의	장애인·비장애인 사용자 대상 시나리오 테스트 및 접근성 지침 준수 여부 평가		
	세부내용	1. 시나리오: 영상통화 번역, 카메라 번역, 사용자사전 등록/적용 2. 평가지표: 작업 성공률≥90%, 학습시간≤10분, SUS 점수, 오류 재시도 용이성 3. 보조기능: 자막 항상표시, TTS 옵션, 큰 버튼(≥44px), 고대비, 경량 UI 모드		

8. 보안 요구사항

요구사항 고유번호		SER-001		
요구사항 명칭		인증 및 권한 보안		
요구사항 분류		보안	응락수준	필수
요구사항 상세설명	정의	사용자 계정 보호와 권한 기반 접근 제어를 통해 보안성을 확보해야 함.		
	세부내용	1. 로그인 시 이메일·전화번호 기반 인증 또는 소셜 로그인(Google, Apple 등) 지원 2. 사용자 역할(Role)을 구분: 일반 사용자 / 관리자 / 통역사 3. 권한 없는 사용자가 수어 사전·번역 로그 등 민감 정보에 접근하지 못하도록 제한 4. 비밀번호는 해시 기반(BCrypt, SHA256 이상)으로 암호화 저장		

요구사항 고유번호		SER-002		
요구사항 명칭		사용자 인터페이스 보안		
요구사항 분류		보안	응락수준	필수
요구사항 상세설명	정의	UI 상에서 개인정보나 민감 데이터가 노출되지 않도록 보안 조치 필요		
	세부내용	1. 사용자 비밀번호 입력은 항상 마스킹 처리 2. 화면에 표시되는 번역 로그/사전은 일정 시간 이후 자동 숨김 또는 삭제 3. 앱 내 특정 화면은 캡처·녹화 방지 기능 적용 4. 소스코드 및 로그에 개인정보가 직접 기록되지 않도록 주석/출력 관리		

요구사항 고유번호		SER-003		
요구사항 명칭		데이터 보안		
요구사항 분류		보안	응락수준	필수
요구사항 상세설명	정의	시스템에 저장되거나 처리되는 데이터의 기밀성·무결성을 보장해야 함.		
	세부내용	1. 사용자 등록 수어, 번역 로그 등은 AES256 수준으로 암호화 저장 2. 실시간 입력 영상은 서버에 저장하지 않고 즉시 처리 후 폐기 3. 사용자 요청 시 모든 데이터는 즉시 삭제 가능 4. DB 접근 권한은 최소 권한 원칙(Least Privilege)에 따라 부여		

요구사항 고유번호		SER-004		
요구사항 명칭		네트워크 보안		
요구사항 분류		보안	응락수준	필수
요구사항 상세설명	정의	클라이언트–서버 및 외부 API 통신 시 안전한 네트워크 연결 보장		
	세부내용	1. 모든 통신은 HTTPS/TLS1.2 이상 적용 2. 번역 API 연동 시 SSL 보안 터널링 또는 VPN 적용 3. 네트워크 접근은 방화벽 및 WAF(Web Application Firewall)로 제어 4. 중간자 공격·재전송 공격 방지 기능 적용		

요구사항 고유번호		SER-005		
요구사항 명칭		외부 서비스 연동 보안		
요구사항 분류		보안	응락수준	필수
요구사항 상세설명	정의	외부 번역 API, 통역사 연결 서비스와의 연동 과정에서 보안 관리 필요		
	세부내용	1. API Key·Access Token은 안전한 별도 저장소에 보관 2. 인증 실패 시 자동 차단 및 관리자 알림 기능 제공 3. 외부 서비스 연동 로그는 주기적으로 감사(Audit) 가능해야 함 4. 외부 연계 시 민감 정보는 전송하지 않고 최소 데이터만 교환		

9. 품질 요구사항

요구사항 고유번호		QUR-001		
요구사항 명칭		신뢰성		
요구사항 분류		품질	응락수준	필수
요구사항 상세설명	정의	시스템이 고장 없이 안정적으로 작동할 수 있는 능력		
	세부내용	1. 평균 무고장 시간(MTBF): 300시간 이상 2. 오류 발생 시 평균 복구 시간(MTTR): 5분 이내 자동 복구 또는 사용자 재시도 유도 3. 실시간 번역 서비스 중 장애 발생 시 즉시 사용자에게 알림 제공		

요구사항 고유번호		QUR-002		
요구사항 명칭		사용성		
요구사항 분류		품질	응락수준	필수
요구사항 상세설명	정의	사용자가 별도 학습 없이도 시스템을 쉽게 이해하고 사용할 수 있는 정도		
	세부내용	1. 장애인을 포함한 사용자 중 90% 이상이 10분 이내에 주요 기능 (영상 번역, 카메라 번역, 사전 등록) 수행 가능해야 함 2. 버튼 크기 ≥ 44px, 고대비 UI, 자막 항상 표시 옵션 제공 3. 사용자 테스트 결과 SUS 점수 70점 이상 달성		

요구사항 고유번호		QUR-003		
요구사항 명칭		접근성		
요구사항 분류		품질	응락수준	보통
요구사항 상세설명	정의	다양한 사용자가 시스템을 불편 없이 이용할 수 있도록 하는 정도		
	세부내용	1. 청각장애인을 위한 자막 필수 제공 2. 시각장애인을 위한 TTS(Text-to-Speech) 옵션 제공 3. 저사양 단말 사용자용 경량 UI 모드 제공 4. 화면 크기·해상도별 반응형 UI 지원		

요구사항 고유번호		QUR-004		
요구사항 명칭		확장성 및 이식성		
요구사항 분류		품질	응락수준	보통
요구사항 상세설명	정의	시스템이 다양한 언어와 환경으로 확장·이식될 수 있는 능력		
	세부내용	1. ASL, JSL 등 다국어 수어 데이터 추가 시 기존 구조 변경 없이 모델 재학습만으로 확장 가능해야 함 2. Android SDK 26+ 호환 필수, iOS 포팅 가능 구조 고려 3. 서버-클라이언트 API는 버전 관리 체계 적용		

요구사항 고유번호		QUR-005		
요구사항 명칭		유지보수성		
요구사항 분류		품질	응락수준	선택
요구사항 상세설명	정의	시스템이 쉽게 수정되고 개선될 수 있는 능력		
	세부내용	1. 기능 단위 모듈화된 코드 구조 (LSTM 모듈, T5 모듈, 번역 API 모듈 등) 2. 기능 수정 시 전체 영향 최소화 → 모듈 단위 테스트 의무화 3. 월 1회 이상 업데이트 계획 수립 및 사용자 피드백 반영		

10. 제약 사항

요구사항 고유번호		COR-001		
요구사항 명칭		데이터 제약		
요구사항 분류		제약사항	응락수준	필수
요구사항 상세설명	정의	학습 및 서비스 운영에 필요한 고품질 수어 데이터 확보의 어려움		
	세부내용	1. 한국 수어 데이터는 AI Hub 공개 자료를 활용할 수 있으나 범위가 제한적임 2. 영어 수어(ASL) 데이터는 고품질 공개 데이터셋 확보가 어려움 3. 비수지기호(표정·시선 등)를 포함한 영상 데이터 수집에 한계 존재 4. 데이터 확보·라벨링 과정에 많은 시간·비용 소요		

요구사항 고유번호		COR-002		
요구사항 명칭		실시간 환경 제약		
요구사항 분류		제약사항	응락수준	필수
요구사항 상세설명	정의	다양한 실환경 조건에서 정확도 저하 발생 가능성		
	세부내용	1. 저조도·복잡한 배경에서 손 랜드마크 인식률 저하 2. 사용자의 빠른 동작이나 손 모양 차이로 인한 정확도 불안정 3. 모바일 단말기의 카메라 성능 차이에 따른 품질 편차 존재		

요구사항 고유번호		COR-003		
요구사항 명칭		모바일 성능 제약		
요구사항 분류		제약사항	응락수준	필수
요구사항 상세설명	정의	저사양 모바일 기기에서의 성능 저하 문제		
	세부내용	1. 프레임 드롭 발생, 번역 지연 가능 2. 발열로 인한 장시간 사용 시 성능 하락 3. 모델 사이즈(T5 등)로 인해 온디바이스 실행이 제한됨 → 서버 의존 필요 4. 최적화를 위해 양자화 모델(TFLite) 또는 경량 모델 사용 필요		

요구사항 고유번호		COR-004		
요구사항 명칭		수어 표준화 제약		
요구사항 분류		제약사항	응락수준	보통
요구사항 상세설명	정의	수어 표현의 지역적·개인적 차이로 인한 번역 일관성 확보 어려움		
	세부내용	1. 동일 의미라도 사용자별 수어 동작 차이 존재 2. 지역·세대별 표현 차이 반영 필요 3. 사용자 맞춤형 사전 기능을 통해 일정 부분 보완 가능		

요구사항 고유번호		COR-005		
요구사항 명칭		운영 비용 제약		
요구사항 분류		제약사항	응락수준	보통
요구사항 상세설명	정의	외부 서비스(API, 전문 통역사 연결) 운영에 따른 비용 발생		
	세부내용	1. 번역 API(Papago) 호출 시 건수 제한·과금 발생 2. 실시간 전문 통역사 연결 기능은 운영 인력 비용 부담 3. 클라우드 서버 운영 비용(GPU/저장소) 지속 발생		

11. 프로젝트 관리 요구사항

요구사항 고유번호		PMR-001		
요구사항 명칭		프로젝트 수행 조직		
요구사항 분류		프로젝트 관리	응락수준	필수
요구사항 상세설명	정의	프로젝트 전체 일정과 품질을 관리하기 위한 조직 구성과 역할 분담을 정의해야 함		
	세부내용	1. 총괄 PM: 프로젝트 일정·품질 총괄 2. AI 모델 담당: 수어 인식(LSTM), 문장 생성(T5) 설계·학습 3. 앱 개발 담당: Android 앱 구현, UI/UX 설계 4. 데이터 담당: 학습용 데이터 수집·정제·전처리 5. QA 담당: 테스트 계획 수립, 사용자 피드백 반영 6. 역할별 책임과 개발 범위를 명확히 구분하여 협업 진행		

요구사항 고유번호		PMR-002		
요구사항 명칭		프로젝트 일정 계획		
요구사항 분류		프로젝트 관리	응락수준	필수
요구사항 상세설명	정의	프로젝트 수행을 단계별 일정과 산출물 중심으로 관리해야 함.		
	세부내용	1. 캡스톤디자인1(전반기): 데이터 수집 및 전처리, AI 모델 개발, 테스트 환경 구축 2. 캡스톤디자인2(후반기): 앱 개발, LSTM/T5 모델 개발 및 최적화 3. 일정 단계: 요구사항 분석 → 설계 → 구현 → 통합 → 테스트 → 보고서 제출 4. 마일스톤: 9월(문장 생성 알고리즘), 10월(번역 API 연동), 11월(앱 통합 구현 및 최종 시연) 5. 산출물: WBS, 마일스톤 관리표, 주간 회의록, 최종 결과 보고서		

요구사항 고유번호		PMR-003		
요구사항 명칭		형상 및 품질 관리		
요구사항 분류		프로젝트 관리	응락수준	보통
요구사항 상세설명	정의	소스코드와 산출물의 형상 관리 및 품질 검증 절차를 수립해야 함.		
	세부내용	1. GitHub 레포지토리 사용, 브랜치 전략(main/dev/feature/*) 적용 2. Pull Request 기반 코드 리뷰 필수화 3. GitHub Actions 통한 자동 빌드·테스트 파이프라인 구축 4. 릴리스 노트 및 태그 관리, 버전별 산출물 기록		

요구사항 고유번호		PMR-004		
요구사항 명칭		리스크 및 변경 관리		
요구사항 분류		프로젝트 관리	응락수준	보통
요구사항 상세설명	정의	프로젝트 수행 중 발생 가능한 위험 요소와 요구사항 변경을 체계적으로 관리해야 함.		
	세부내용	1. 위험 식별: 데이터 부족, 성능 저하, 일정 지연, 외부 API 오류 2. 대응 방안: 데이터 보강 계획, 경량화 모델 적용, 일정 재조정, API 폴백 설계 3. 변경 관리: 요구사항 변경 발생 시 회의록 기록 후 GitHub Issue/Project Board에 등록 4. 승인 절차: 팀 내 합의 → 교수님 승인 → 반영		