이브와

SW개발/HW제작 설계서

프로젝트 명 :
KoBERT Machine Learning 기반의
A.I 보이스 피싱 예방

수행 단계별 주요 산출물

단계	산출물	일반 ·모바일 APP ·Web 등	응용 소프트웨어 ·빅데이터 ·인공지능 ·블록체인 등	응용 하드웨어 ·IoT ·로봇 ·드론 등
환경 분석	시장/기술 환경 분석서	△	△	△
	설문조사 결과서	△	△	△
	인터뷰 결과서	△	△	△
요구사항 분석	요구사항 정의서	○	○	○
	유즈케이스 정의서	△	△	△
아키텍처 설계	서비스 구성도(시스템 구성도)	○	○	○
	서비스 흐름도(데이터 흐름도)	△	○	△
	UI/UX 정의서	△	△	△
	하드웨어/센서 구성도	-	-	○
기능 설계	메뉴 구성도	○	○	○
	화면 설계서	○	○	△
	엔티티 관계도	○	○	△
	기능 처리도(기능 흐름도)	○	○	○
	알고리즘 명세서/설명서	△	○	○
	데이터 수집처리 정의서	-	○	-
	하드웨어 설계도	-	-	○
개발 / 구현	프로그램 목록	○	○	○
	테이블 정의서	○	○	△
	핵심 소스코드	○	○	○

※ ○ 권장, △ 선택

데일리안

금융사기, 생애주기 특징 악용..."고령층 메신저 피싱에 취약"

보이스피싱 등 금융사기 수법이 생애주기를 악용하며 더욱 고도화되고 있다는 분석이 나왔다. 특히 고령층은 가족이나 지인을 사칭한 메신저 피싱에...

2022. 8. 11.

연합뉴스 PiCK | 2023.05.27. | 네이버뉴스

[진화하는 **보이스피싱**] ① 누구든 당할 수 있다...연간 피해 5천억

연합뉴스는 피해자에 대한 '경제적 살인'이라고 불리는 **보이스피싱**의 **심각성**을 짚어보고, 근절 방안을 제시하는 기획 기사 3편을 송고합니다. (수원=연합뉴스) 강...

위 기사에서 보다시피 보이스피싱 수법은 나날이 진화하고 있는 상황이다. 현존하는 보이스피싱 예방 앱을 보면, 스마트폰 분야에 대해서는 이미 활성화되어있지만 **인터넷 전화나 유선 전화와 같은 분야에서는 매우 생소하다는 한계점** 을 찾을 수 있다.

이에 우리는 일부 보이스피싱 취약계층을 위한 저렴한 비용의 대화 내용을 실시간으로 분석하여, 보이스피싱 여부를 가려내기 위한 시스템을 개발하고자 한다.

시장/기술 동향 분석

- **기존 제품과의 차별성**

측정지표	시중에 나와있는 피싱 예방 관련 서비스	본 프로젝트
기능성	대부분 스마트폰, 즉 무선 전화에서 피싱 탐지가 가능한 서비스임 따라서 스마트폰 사용이 익숙치 않은 노인계층들이 보이스피싱을 탐지하는데에 많은 어려움이 있을 거라 예상이 됨.	스마트폰이 아닌 유선 전화를 타켓으로 하여 보통의 집 전화에서도 바로 피싱 탐지가 가능하게끔 설정
사용성	주 타켓이 비장애인이다보니 장애인이 사용하기에는 서비스적으로 많은 불편함이 존재할 것으로 예상이 됨	- 청각장애인을 위하여 불빛으로 피싱 여부를 알려줌 - 또한 시각장애인을 위하여 음성으로도 피싱 여부를 알려줌

| 요구사항 정의서 (1)

요구사항 ID	요구사항명	기능 ID	기능명	세부사항	예외사항
A01	보이스피싱 탐지 결과 알림 기능	A01_B01	피싱 여부 음성 알림 기능	보이스피싱 탐지 알고리즘을 거쳐서 피싱 결과에 대해 스피커를 통해 음성 알림 기능으로 제공할 수 있도록 한다.	
		A01_B02	LED를 이용한 피싱 알림 기능	보이스피싱 탐지 알고리즘을 거쳐서 피싱여부에 대해 LED를 이용해 위험, 주의, 안전으로 시각적인 효과로 알림을 제공할 수 있도록 한다.	

| 요구사항 정의서 (2)

구분	기능	설명
S/W	STT 기능 및 전처리 알고리즘	녹취된 음성을 AWS transcribe API를 통해 STT를 수행하고 알고리즘을 통해 변환된 텍스트에 대한 전처리를 수행하여 딥러닝 모델로 전송한다.
	보이스피싱 탐지 모델	전처리된 텍스트 데이터를 사용하여 pre-trained 한국어 NLP 오픈소스 모델을 파인튜닝하여 보이스피싱 여부에 대한 다중 분류를 수행한다.

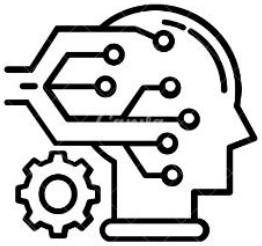
구분	기능	설명
H/W	라즈베리 파이를 이용한 데이터 수집기 구현	Raspberry Pi 에 부착한 마이크를 통해서 녹취를 수행하고 녹음된 파일을 AWS 서버로 전송한다.
	라즈베리 파이를 이용한 피싱 여부 음성 알림 기능	모델 결과를 받아와 보이스피싱 다중 분류에 대한 결과를 라즈베리파이에 부착된 스피커를 통해 음성 알림 기능으로 제공할 수 있도록 한다.
	라즈베리파이 LED 를 이용한 피싱 알림 기능	모델 결과를 받아와 보이스피싱 다중 분류에 대한 결과를 라즈베리파이에 부착된 LED를 이용해 위험, 주의, 안전으로 시각적인 효과로 알림을 제공할 수 있도록 한다.

| 서비스 구성도 – 서비스 시나리오

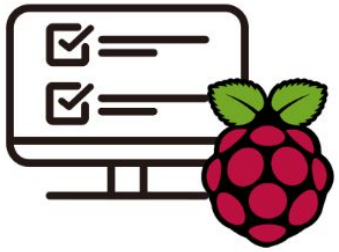
1. 통화 시작 시 음성 인식 시작 →

2. AWS 서버로 통화 내역 전송 →

3. STT로 데이터 변환



4. 변환 내용 머신러닝 분류 →

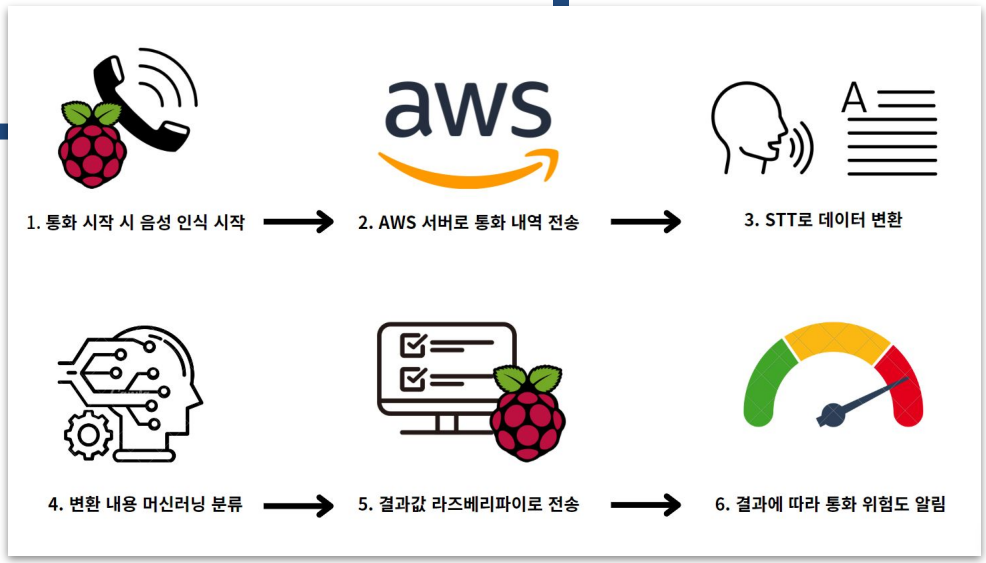


5. 결과값 라즈베리파이로 전송 →

6. 결과에 따라 통화 위험도 알림

| 서비스 구성도 – 서비스 시나리오

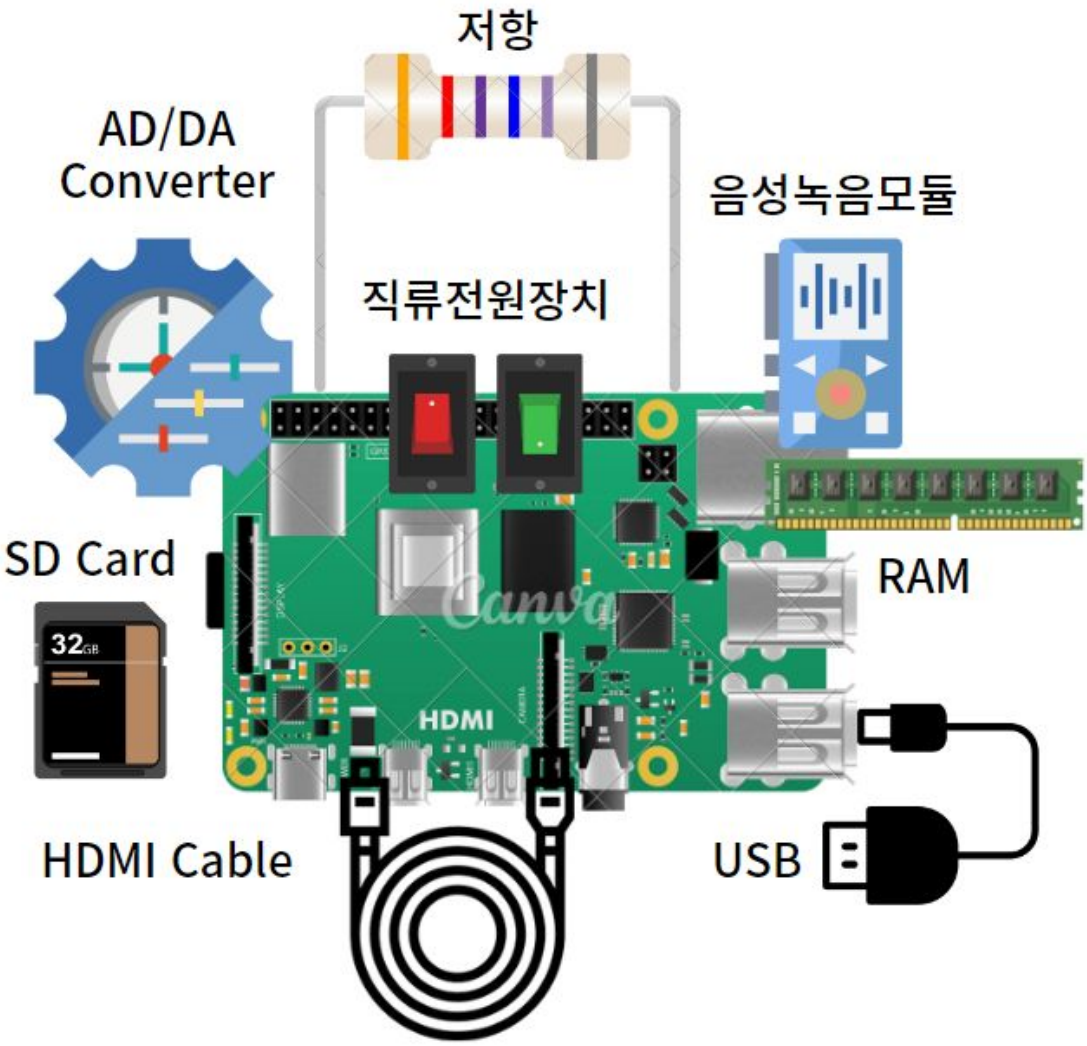
1. 전화 통화가 시작되면 라즈베리파이가 음성 인식 시작
2. 전화 통화 중 통화 내역을 실시간으로 AWS Transcribe 서버로 전송
3. 전송된 데이터를 STT 기술을 통해 변환
4. 변환된 내용을 딥러닝을 통해 분류
5. 분류 작업 결과값을 다시 라즈베리파이가 받아옴
6. 받아온 결과에 따라 통화의 위험도 알림



| 하드웨어 / 센서 구성

구성품명	분류	설명
Gravity: Voice Recorder Module Pro	모듈	라즈베리파이와 연결할 음성 녹음용 모듈
Solderless Breadboard	브레드보드	라즈베리 파이 연결용 보드
Jumper Cable	케이블	부품들을 연결하기 위해 사용하는 케이블
NX-HD20015-MICRO(NX496)	케이블	연결을 위한 1.5M Micro HDMI 케이블 2.0 Ver
Sandisk Ultra microSDHC UHS-I Card	메모리 카드	데이터 관리를 위한 메모리 카드
DZ018CHI_050300K	전원장치	전원 공급을 위한 직류전원장치
Raspberry Pi 4 Computer Model B	저장장치	데이터 저장을 위한 4GB RAM
Card Reader USB 2.0	저장장치	데이터 이동과 처리를 위한 리더 USB
Raspberry Pi-4 Premium Case	물리적 저장장치	라즈베리파이 저장용 케이스
막대 저항(10/560 오메가 각 5EA)	기타 부품	저항 처리용 막대
Tact 스위치	스위치	전원 차단용 스위치
AD/DA 컨버터 모듈	모듈	아날로그와 디지털 간 신호 변환을 위한 모듈

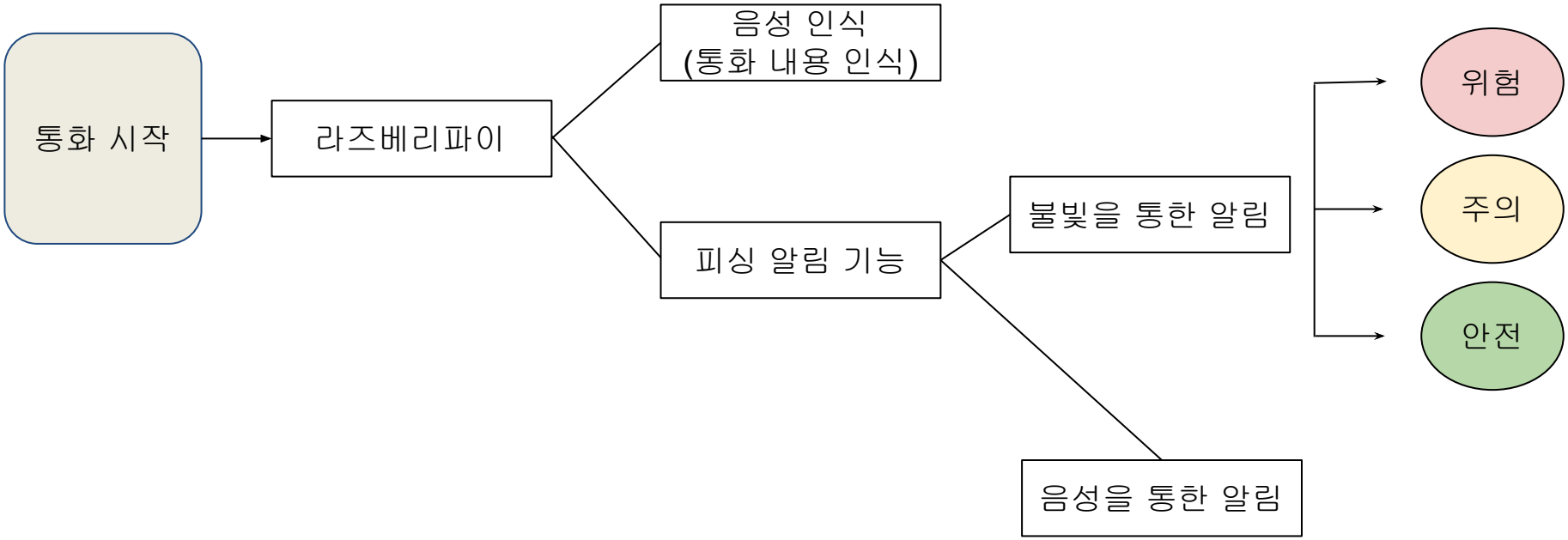
| 하드웨어 / 센서 구성도



기타 삽입 사항	
구성품명	분류 & 상세
구리방열판	방열용 구리판
LED_01	알림용 녹색 LED
LED_02	알림용 적색 LED
LED_03	알림용 황색 LED
캐패시터	노이즈 필터링용 캐패시터
네오박스	물품 보호용 장치
GPIO	입출력 제어를 위한 핀

| 메뉴 구성도

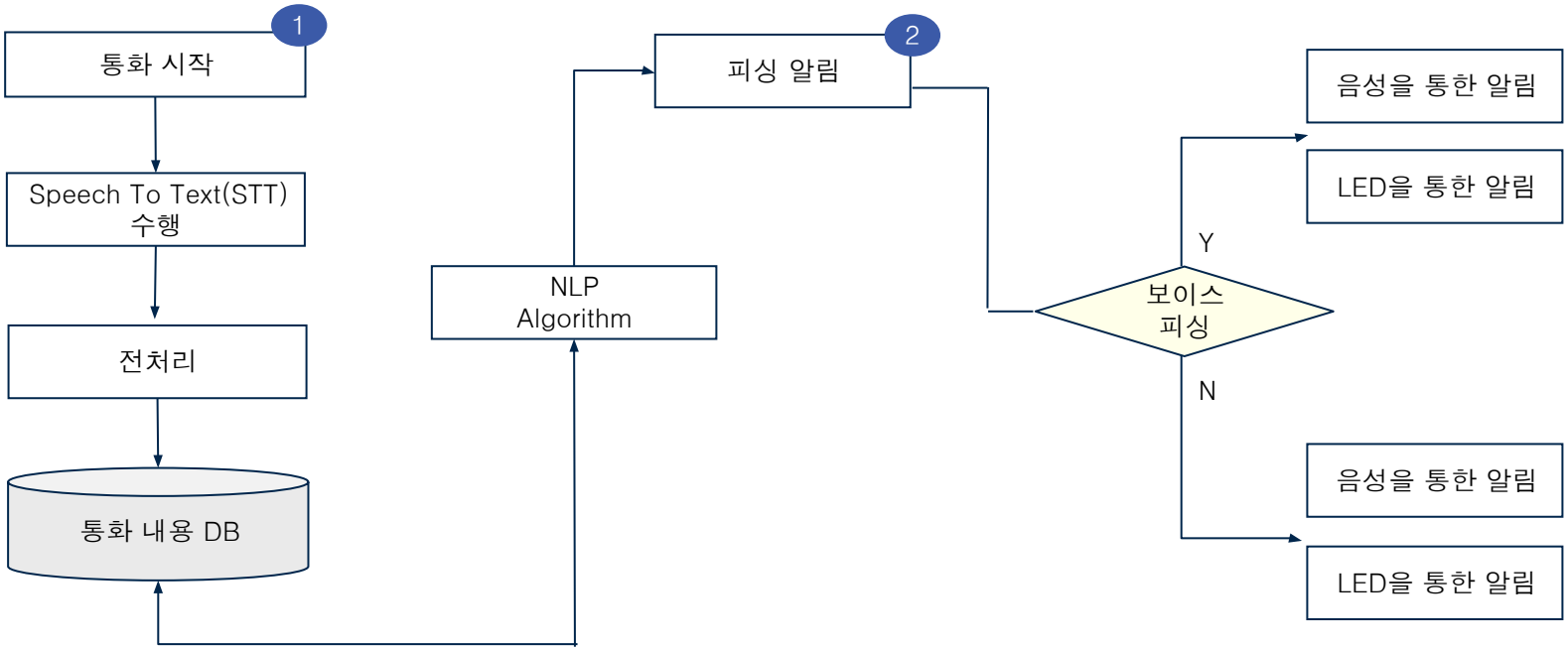
- **메뉴 구성도**



| 기능 처리도(기능 흐름도)

프로그램 ID	23_IF001	프로그램 명	피싱 탐지 프로그램	작성일	2023.06.25	Page	-
개요	라즈베리파이 모듈 및 딥러닝 기반 자연어처리 모델 활용한 사회적 약자 및 피싱 취약계층을 위한 보이스피싱 탐지 서비스					작성자	피라냐팀

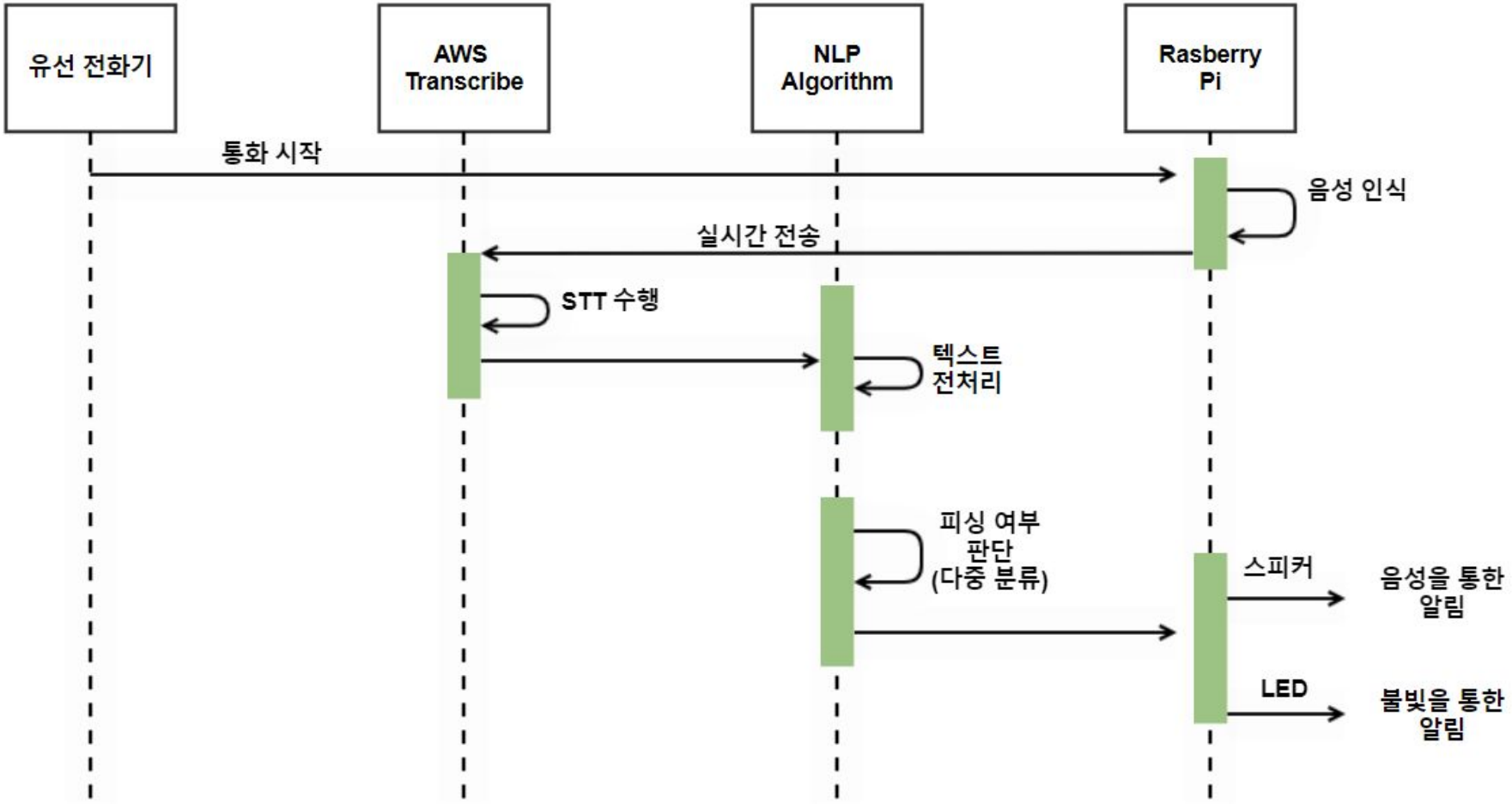
<자연어 처리를 활용한 보이스피싱 탐지 서비스 기능 흐름도> 기능 흐름도



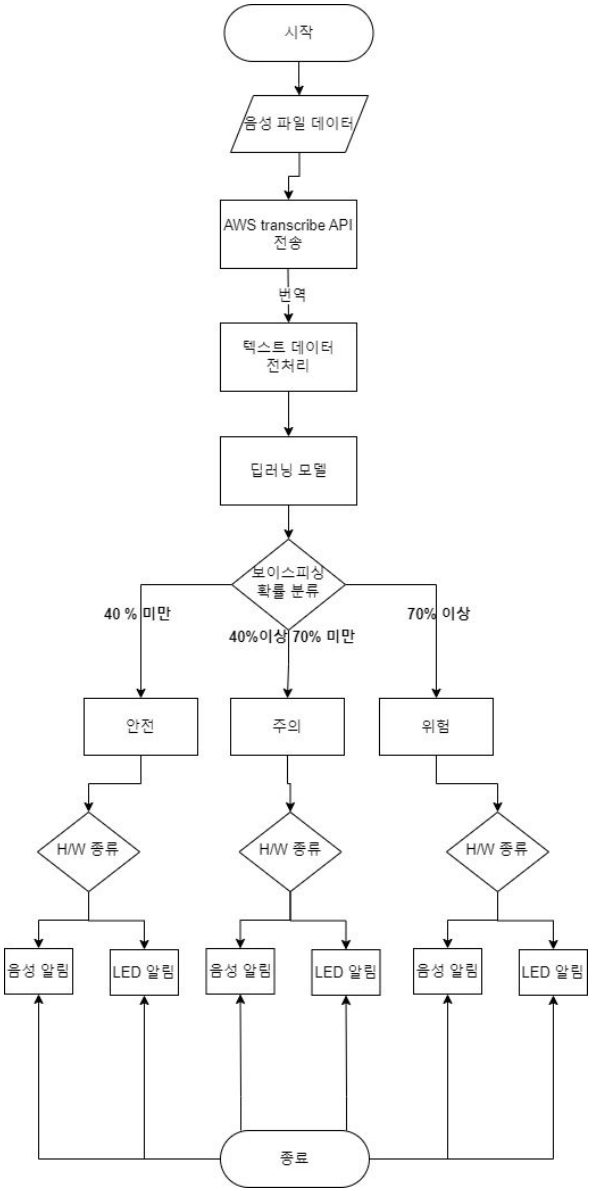
1 수신자가 송신자와 통화를 시작하면 AWS Transcribe가 실시간으로 통화내용을 수집하여 STT를 수행한다. 그 후 자연어 처리 모델 알고리즘을 활용하여 전처리를 수행하여 DB에 저장하면, 저장된 DB를 이용하여 피싱 알림 기능을 제공한다.

2 통화 내용 DB를 바탕으로 사전에 Fine-Tuning된 NLP Algorithm이 피싱 여부에 대해서 다중 분류를 진행한다. 그리고, 최종적으로 이 결과를 바탕으로 사용자에게 피싱 여부를 라즈베리파이(스피커, LED)를 활용하여 음성 및 불빛으로 알려준다.

| 기능 처리도(기능 흐름도)

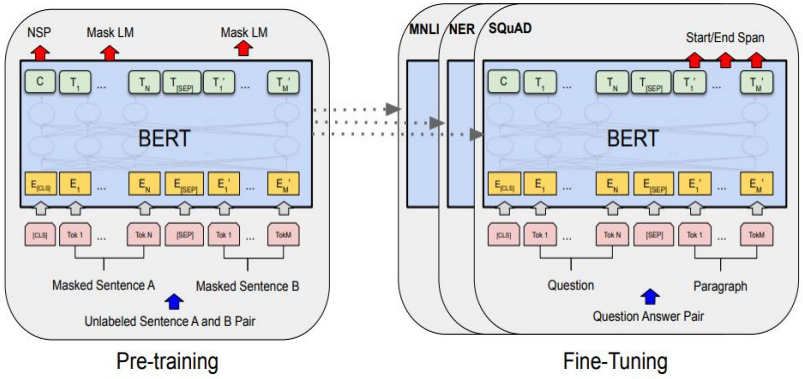

유선 전화기
AWS Transcribe
NLP Algorithm
Rasberry Pi
통화 시작
음성 인식
실시간 전송
STT 수행
텍스트 전처리
피싱 여부 판단 (다중 분류)
스피커
음성을 통한 알림
LED
불빛을 통한 알림

| 알고리즘 명세서



*** 음성데이터 처리를 통한 보이스피싱 탐지 알고리즘**

1. 수집한 음성파일데이터를 AWS transcribe API로 전송한다.
2. AWS를 통해 번역을 수행하고 해당 파일을 통해 얻은 텍스트 데이터를 전처리 한다.
3. 전처리 완료된 텍스트 데이터를 바탕으로 적합한 모델을 통해 피싱 여부에 대한 다중분류를 수행한다
4. 보이스피싱 확률 분류가 40 % 미만인 경우는 안전, 40% 이상 70 % 미만인 경우는 주의, 70% 이상인 경우는 위험으로 분류한다.
5. 해당 결과를 바탕으로 음성 알림과 LED 알림 서비스를 수행한다.
6. 모든 서비스가 수행되면 종료한다.



- **사용할 알고리즘 : KoBERT**

KoBERT 는 한국어 자연어처리를 위해 개발된 모델 입니다.

BERT 알고리즘을 기반으로 구성되어 있습니다.

1. 사전 훈련 단계

한국어에 특화된 텍스트 데이터로 사전 훈련되어 한국어의 언어적 특성과 문맥을 파악할 수 있도록 합니다.

이를 통해 한국어 문장에서 의미있는 정보를 추출하고 다양한 자연어 처리 작업을 수행할 수 있습니다.

2. 파인 튜닝 단계

사전 훈련된 모델을 사용해 파인튜닝하면 각각의 테스크에 맞게끔 모델을 이용할 수 있습니다.

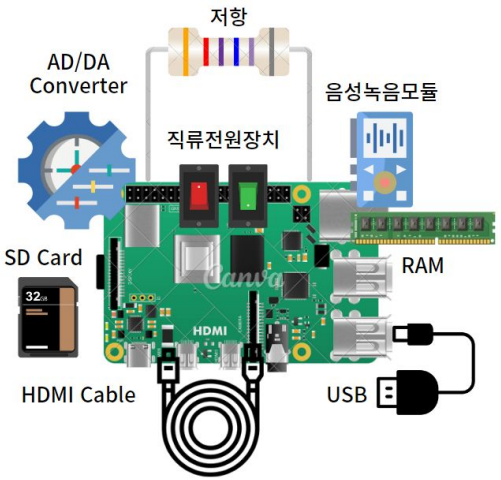
KoBERT의 경우 문장의 의미를 이해하고 감정 분석, 질문 분석, 문장 분류, 개체명 인식과 같은 작업을 수행할 수 있습니다.

여기서 저희는 문장분류의 부분을 사용할 예정입니다.

| 하드웨어 설계도

구성품명	분류	설명
Gravity: Voice Recorder Module Pro	모듈	라즈베리파이와 연결할 음성 녹음용 모듈
Solderless Breadboard	브레드보드	라즈베리 파이 연결용 보드
Jumper Cable	케이블	부품들을 연결하기 위해 사용하는 케이블
NX-HD20015-MICRO(NX496)	케이블	연결을 위한 1.5M Micro HDMI 케이블 2.0 Ver
Sandisk Ultra microSDHC UHS-I Card	메모리 카드	데이터 관리를 위한 메모리 카드
DZ018CHI_050300K	전원장치	전원 공급을 위한 직류전원장치
Raspberry Pi 4 Computer Model B	저장장치	데이터 저장을 위한 4GB RAM
Card Reader USB 2.0	저장장치	데이터 이동과 처리를 위한 리더 USB
Raspberry Pi-4 Premium Case	물리적 저장장치	라즈베리파이 저장용 케이스
막대 저항(10/560 오메가 각 5EA)	기타 부품	저항 처리용 막대
Tact 스위치	스위치	전원 차단용 스위치
AD/DA 컨버터 모듈	모듈	아날로그와 디지털 간 신호 변환을 위한 모듈

구성품명	분류 & 상세
구리방열판	방열용 구리판
LED_01	알림용 녹색 LED
LED_02	알림용 적색 LED
LED_03	알림용 황색 LED
캐패시터	노이즈 필터링용 캐패시터
네오박스	물품 보호용 장치
GPIO	입출력 제어를 위한 핀



기본적으로 비치된 Raspberry Pi Bread Board에 저항Unit, AD/DA Converter을 우선적으로 부착한다. AD/DA Converter의 경우 본 프로젝트가 음성과 관련된 data를 다루는 만큼, 주변의의 noise 제거에 필요하여 추가하였다. 그 외에 data saving을 위한SD Card, USB 카드도 부착한다. 또 외부 단자와 연결하기 위한DHMI Cable, 전원 공급 제어를 위한**직류전원장치**도 추가적으로 부착한다. 마지막으로 위험도와 관련된 알림 기능을 구현하기 위해 3가지 색상의LED와 **음성 인식 및 녹음 모듈**을 부착한다. 이 외에도GPIO, Capaciter을 부착하고 구리 방열판도 이용하여 HardWare Structure을 완성하였다.

| 프로그램 – 목록

기능 분류		기능번호	기능 명
ALERT	VOALERT	VOALERT-01-01	녹색 시각 신호 표시
		VOALERT-01-02	적색 시각 신호 표시
		VOALERT-01-03	노란색 시각 신호 표시
	COALERT	COALERT-01-04	위험도 '안전' 상태 알림
		COALERT-01-05	위험도 '주의' 상태 알림
		COALERT-01-06	위험도 '위험' 상태 알림
SORT		SORT-02-01	위험도 '안전' 상태 & 단어 분류
		SORT-02-02	위험도 '주의' 상태 & 단어 분류
		SORT-02-03	위험도 '위험' 상태 & 단어 분류
SEND		SEND-01-01	통화 내역 전송
GET		GET-01-01	분류 결과값 수신

| 핵심소스코드(1-1)

```
1   #!/usr/bin/env python
2   # coding: utf-8
3
4   # In[34]:
5
6
7   from selenium import webdriver
8   from selenium.webdriver import ActionChains
9   from selenium.webdriver.common.keys import Keys
10  from selenium.webdriver.common.by import By
11  import time
12  import requests
13  from bs4 import BeautifulSoup
```

- 음성 데이터 수집을 위한 Web 크롤링 스크립트 전문

1. selenium 패키지에서 필요한 모듈들을 import

webdriver	웹 브라우저 제어를 위한 클래스
ActionChains	마우스 및 키보드 동작 모방을 위한 클래스
Keys	특수 키 제어를 위한 클래스
By0	웹 요소를 찾기 위한 방법 지정을 위한 클래스

2. 일시적인 딜레이를 주기 위한 time 모듈 import

3. 웹 페이지의 내용을 가져오기 위한 requests 모듈 import

4. HTML & XML 문서 파싱과 검색을 위한 BeautifulSoup 라이브러리 가져오기

| 핵심소스코드(1-2)

```
19  browser = webdriver.Chrome('./chromedriver.exe')
20  url = 'https://www.fss.or.kr/fss/bbs/B0000207/list.do?menuNo=200691&bbsId=&cl1Cd=&pageIndex=16&sdate=&edate=&searchCnd=1&searchWrd='
21  browser.get(url)
```

5. webdriver 모듈에서 Chrome 클래스를 사용하여 Chrome 브라우저를 제어하기 위한 드라이버를 생성

6. ./chromedriver.exe는 Chrome 드라이버의 파일 경로

--> 해당 파일은 Chrome 브라우저와 Selenium이 상호 작용할 수 있도록 해주는 역할을 함

7. url 변수에 스크래핑하고자 하는 웹 페이지의 URL값 저장

8. browser 객체의 get 메서드를 사용하여 브라우저를 해당 URL로 이동

--> 이를 통해 Selenium이 브라우저를 제어하고, 지정된 URL에 접속할 수 있게 됨

```
act = ActionChains(browser)
lst = []
```

9. ActionChains 클래스의 browser 객체를 이용하여 act 객체 생성

--> ActionChains Class

: 브라우저에서 수행할 동작들을 정의하고 제어하기 위한 클래스

10. 음성 파일의 URL을 저장할 리스트인 lst 생성

| 핵심소스코드(1-3)

```
from selenium.webdriver.common.by import By

for i in range(1, 10):
    try:
        element = browser.find_element(By.CSS_SELECTOR, f'#content > div.bd-list > table > tbody > tr:nth-child({i}) > td.title > a')
        act.click(element).perform()
        time.sleep(2)
        target_element = browser.find_element(By.CSS_SELECTOR, '#content > div.bd-view > div > video')
        audio_url = target_element.get_attribute("src")
        lst.append(audio_url)
    except NoSuchElementException:
        print(f"Element not found for row {i}. Skipping...")
    finally:
        browser.back()
        time.sleep(2)
```

11. for 루프를 사용하여 특정 범위의 행에 대해 아래 행위를 반복
    - find_element 메서드를 통해 CSS 선택자로 웹 요소 탐색
    - act.click(element).perform() 을 통해 찾은 요소 클릭
    - time.sleep(2) 를 통해 2초 동안 대기
    - target_element.get_attribute("src")를 사용하여 동영상 요소의 속성에서 음성 파일 URL 가져오기
    - 가져온 URL을 lst 리스트에 추가
    - browser.back()을 통해 이전 페이지로 되돌아가기
    - time.sleep(2) 를 통해 2초 동안 대기

+ 만약 NoSuchElementException 예외가 발생할 경우, 해당 행이 없다는 메시지를 출력하고 해당 순서를 스킵함

+ 11번 코드까지 실행했을 경우, 아래와 같이 브라우저가 지속적으로 제어된다.

| 핵심소스코드(1-4)

12. print(lst)를 통해 수집한 음성 파일의 URL 출력

```
['https://www.fss.or.kr/fss/cmmn/file/fileDown.do?menuNo=200691&atchFileId=d85ea38e9d666e0e002595207abbc813&fileSn=1&bbsId=B0000207',
'https://www.fss.or.kr/fss/cmmn/file/fileDown.do?menuNo=200691&atchFileId=a5304c603a20b5489481388d5574e9c6&fileSn=1&bbsId=B0000207',
'https://www.fss.or.kr/fss/cmmn/file/fileDown.do?menuNo=200691&atchFileId=8514ee9158c7716717a76cb06187ccee&fileSn=1&bbsId=B0000207',
'https://www.fss.or.kr/fss/cmmn/file/fileDown.do?menuNo=200691&atchFileId=1dbd7e3c6871ea96910ed578d749ec81&fileSn=1&bbsId=B0000207',
'https://www.fss.or.kr/fss/cmmn/file/fileDown.do?menuNo=200691&atchFileId=ad081d750a82b0bbac1b8c70ce7b8143&fileSn=1&bbsId=B0000207',
'https://www.fss.or.kr/fss/cmmn/file/fileDown.do?menuNo=200691&atchFileId=3002573c73d3c96dcbe44eecbce461c7&fileSn=1&bbsId=B0000207',
'https://www.fss.or.kr/fss/cmmn/file/fileDown.do?menuNo=200691&atchFileId=15a28094ba988502175b4a4bc0a5bd08&fileSn=1&bbsId=B0000207',
'https://www.fss.or.kr/fss/cmmn/file/fileDown.do?menuNo=200691&atchFileId=4a1c828374a16332471965416lecc84c&fileSn=1&bbsId=B0000207',
'https://www.fss.or.kr/fss/cmmn/file/fileDown.do?menuNo=200691&atchFileId=3ca28d22754c9676b608152dd96b4d52&fileSn=1&bbsId=B0000207']
```

13. save_directory 변수에 음성 파일을 저장할 경로를 지정

+ 만일 경로가 존재하지 않을 경우 path 생성

14. for 루프를 사용하여 lst에 저장된 음성 피일의 URL에 대해 아래 과정 반복

- requests.get(url)을 사용하여 해당 URL로 GET 요청 전송
- 수령한 응답을 response 변수에 저장
- file_path 변수에 저장할 파일명과 저장 경로를 지정
- os.path.join() 함수를 이용하여 save_directory 와 파일명(변수 i 의 값)을 결합
- open(file_path, 'wb')를 사용하여 파일을 쓰기 모드로 Open --> 'wb'은 파일을 이진 모드로 열기 위한 플래그
- file.write(response.content)를 사용하여 응답의 내용을 파일에 작성

15. 모든 작업이 끝나면 browser.quit() 메서드를 통해 Selenium WebDriver 종료

```python
# Print the collected URLs
print(lst)


i = 1
for url in lst:
    response = requests.get(url)
    with open(f'{i}.mp3','wb') as file:
        file.write(response.content)
        i+=1

browser.quit()
```

| 핵심소스코드(2)

- 음성파일을 AWS Transcribe 사용해 STT 하는 소스코드
- 해당 코드의 결과로 AWS 서버에 transcribe한 결과 저장

```
from __future__ import print_function
import time
import boto3

url_lst = []

transcribe = boto3.client("transcribe")
for i in range(88,117):
    job_name = f"transcribes..{i}"
    job_url = f"s3://piranah-transcribe-bucket/수사기관 사칭형/{i}.mp3"
    transcribe.start_transcription_job(
        TranscriptionJobName = job_name,
        Media = {"MediaFileUri": job_url},
        MediaFormat = "mp3",
        LanguageCode = 'ko-KR'
    )

    while True:
        status = transcribe.get_transcription_job(TranscriptionJobName = job_name)
        if status["TranscriptionJob"]["TranscriptionJobStatus"] in ['COMPLETED','FAILED']:
            break
        print("Not ready yet...")
        time.sleep(5)
    print(status)
```

| 핵심소스코드(3)

- AWS Transcribe 저장 자동화 코드(일부)
  - 1페이지에 있는 파일 저장

```
#json파일 다운로드


#첫페이지
for i in range(1,11):
    time.sleep(2)
    #Json파일 클릭
    try:
        browser.find_element(By.XPATH,f'//*[@id="transcription-job-table"]/div[2]/div[1]/table/tbody/tr[{i}]/td[1]/label').click()
        time.sleep(2)
    except:
        browser.find_element(By.XPATH,f'//*[@id="transcription-job-table"]/div[2]/div[1]/table/tbody/tr[{i}]/td[1]/label').send_keys(Keys
        time.sleep(2)

    #다운로드 클릭
    try:
        browser.find_element(By.XPATH,'//*[@id="job-list-download-job"]').click()
        time.sleep(2)
    except:
        browser.find_element(By.XPATH,'//*[@id="job-list-download-job"]').send_keys(Keys.ENTER)
        time.sleep(2)

    #바뀔 Json파일 이름
    file_name=browser.find_element(By.XPATH,f'//*[@id="transcription-job-table"]/div[2]/div[1]/table/tbody/tr[{i}]/td[2]/a')
    newfile_name=file_name.text

    #다운로드 한 Json파일의 현재 경로 지정
    filepath = 'C:₩₩Users₩₩samsung₩₩Desktop₩₩2023 이브와 프로젝트₩₩KoBERT Machine Learning 기반의 A.I 보이스 피싱 예방₩₩음성 파일 크롤링
    filename = max([filepath + '₩₩' + f for f in os.listdir(filepath)], key=os.path.getctime)
    time.sleep(2)

    #Json파일 이름 변경
    shutil.move(filename,os.path.join(filepath, newfile_name+'.json'))
```

Thank you