수집 데이터 및 데이터 전처리 보고서

2025년 **11**월 **17**일

1. 문서 개요

본 문서는 “음성에서 문서로, *AI*가 만드는 회의 자동화 시스템” 구축 과정에서 시스템 성능 검증을 위해 수집한 텍스트 데이터의 구조와 전처리 과정을 기술하기 위해 작성되었다.

본 프로젝트는 음성 인식 모델인 Whisper와 화자분리 모델 Pyannote를 활용하여 실제 회의 음성을 텍스트로 전사하고, 이를 기반으로 요약, 태스크 자동 추출, 이슈·안건 도출 기능을 수행하는 사내용 AI 시스템을 개발하는 것을 목표로 한다. 따라서 본 문서에서 다루는 데이터는 다음과 같은 목적에 사용된다.

- Whisper 전사 품질 평가

- Pyannote 화자 분리 정확도 검증

- 요약·태스크 추출 모델의 실제 회의 적용성 검증

- 기능별 시스템 시나리오 테스트

2. 데이터 수집 방법

본 프로젝트의 시스템 평가를 위해 수집된 회의 음성 및 스크립트 데이터는 Whisper 기반 음성 인식, Pyannote 화자 분리, 요약·태스크 추출 기능의 성능을 종합적으로 검증하는 데 활용된다.

구분	데이터 유형	출처	수집량	용량	비고
음성	회의 음성 파일	YouTube – 테디노트	20개	1 GB	외부 IT 회의
텍스트	회의 원본 스크립트	YouTube – 테디노트	20개	6 MB	외부 IT 회의
음성	사내 회의 녹음 파일	팀 내부 회의	17개	1.65 GB	실제 회의 환경
텍스트	사내 회의 스크립트	내부 전사 앱	17개	241 KB	Whisper와 비교
텍스트	시나리오 기반 회의 스크립트	OpenAI GPT 모델 생성	12개	1 MB	생성 IT 회의
텍스트	파인튜닝용 용어 사전	(한국정보통신기술협회)	41,967개	18 MB	IT 용어 크롤링
텍스트	파인튜닝용 용어 사전	(한국정보통신기술협회)	79개	19 KB	IT 용어 pdf 추출
텍스트	파인튜닝용 용어 사전	OpenAI GPT 모델 생성	744개	123 KB	판교어 정의쌍
텍스트	파인튜닝용 용어 사전	크몽 - 사회초년생 및 IT 입문자를 위한 판교어 300선	300개	317 KB	판교어 pdf 추출

2.1. YouTube 기반 IT 회의 영상 데이터 추출

- 출처: YouTube 채널 테디노트(TeddyNote)
- 수집 목적: 외부 IT 실무 맥락을 포함하기 위해 회의 및 기술 설명 영상을 20건 수집하여 음성과 스크립트를 확보. Whisper 전사 결과와 비교하여 STT 품질을 평가하는 데 활용.
- 수집 전략:
    1. YouTube API를 활용하여 영상 메타데이터 수집
    2. 영상 다운로드 후 ffmpeg 기반 음성 추출
    3. YouTube 자막(SRT/TXT) 또는 자동 자막 확보
- 수집량: 음성 20건 + 스크립트 20건 (총 40건)
- 파일명 예시: 영상명_크리에이터명.*mp3*
  (예시: Anthropic에서 공개한 RAG 성능 올리는 팁!_테디노트.*mp3*)
  영상명_크리에이터명.*txt*
  (예시: Anthropic에서 공개한 RAG 성능 올리는 팁!_테디노트.*txt)*
- 수집 코드(Python)
  1) 동영상 메타데이터 수집

```python
def list_all_videos_with_duration(channel_url: str):
    opts = dict(YDL_BASE_OPTS)
    with YoutubeDL(opts) as ydl:
        info = ydl.extract_info(channel_url, download=False)
    entries = (info or {}).get("entries") or []
    if not isinstance(entries, list):
        entries = [info]

    out = []

    with YoutubeDL(opts) as ydl:
        for e in entries:
            if not e or not e.get("id"):
                continue
            url = e.get("webpage_url") or e.get("url") or f"https://www.youtube.com/watch?v={e['id']}"
            inf = ydl.extract_info(url, download=False)
            if not inf:
                continue
            out.append({
                "id": inf.get("id"),
                "title": inf.get("title") or e.get("title"),
                "url": inf.get("webpage_url") or url,
                "duration": inf.get("duration"),
            })
    return out
```

  2) 음성, 자막 수집

```python
kept = 0
for i, v in enumerate(videos, 1):
    if kept >= NEED_N:
        break

    vid = v["id"]
    dur = v.get("duration") or 0
    if dur < MIN_SECONDS:
        # 30분 미만은 스킵
        print(f"[{i}] SKIP (duration {dur}s < 1800s): {v.get('title')}")
        continue

    title = sanitize(v["title"] or vid)
    print(f"\n[{i}] CHECK {title} ({vid}) | duration={dur}s")

    # 자막 유무 확인
    txt = get_transcript_text(vid, LANGS)
    if not txt:
        print("  - 자막 없음 → 스킵")
        continue

    # 스크립트/오디오 동일 제목 저장
    base_no_ext = os.path.join(OUT_DIR, title)
    base_no_ext = uniquify(os.path.join(OUT_DIR, title))

    # 스크립트(.txt) 저장
    tr_path = base_no_ext + ".txt"
    with open(tr_path, "w", encoding="utf-8") as f:
        f.write(txt)
    print(f"자막 저장: {tr_path}")

    # 오디오 저장
    download_audio(v["url"], base_no_ext)
    print(f"오디오 저장: {base_no_ext}.[{AUDIO_FMT}]")
```

2.2. 사내 실회의 음성 녹음 데이터 수집

- 출처: 내부 팀회의
- 수집 목적: 실제 프로젝트 회의를 직접 녹음한 음성을 기반으로 Whisper 전사 정확도 및 화자 분리 성능 검증 위함
- 수집 전략:
    1. 사내 회의를 스마트폰/PC 녹음 파일을 이용하여 직접 녹음
    2. 외부 전사 애플리케이션(다글로, 에이닷, 크로바, 티로)을 활용하여 스크립트 추출
        a. 다글로, 에이닷, 크로바, 티로 전사 스크립트 기반 적합성 판단 후 화자 맵핑 진행
- 수집량: 음성 5건 + 스크립트 5건
- 파일명 예시: meeting_수집연월일_수집어플명_audio.*m4a*
  *(*예시:meeting_20250115_에이닷_audio.*m4a)*
  meeting_수집연월일_수집어플명_script.*txt*
  *(*예시:meeting_20250115_에이닷_script.*txt)*

2.3. OpenAI 기반 시나리오 생성 회의록 스크립트 생성

- 수집 방법: OpenAI GPT 모델을 활용하여 회의 시나리오 스크립트 생성
- 수집 목적: 실제 IT 기업에서 진행하는 회의 상황을 반영하기 위해 고품질 회의 스크립트를 추가로 생성하여 Whisper 전사 품질 검증 및 요약·태스크 추출, 잡담 필터링 성능 테스트에 활용
- 특징:
    1. 화자 구분 포함
    2. 불필요한 잡담 포함
    3. 태스크 / 세부 안건 포함
- 수집량: 스크립트 12건
- 파일명 예시: synthetic_meeting_소요시간.*txt*
  (예시: synthetic_meeting_15min.*txt*)
- 수집 코드(Python):

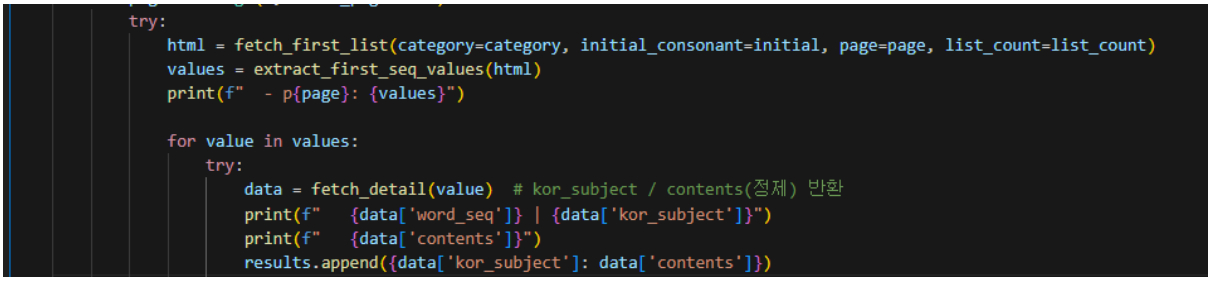
```
def generate_meeting_script(duration_min: int):

    prompt = f"""
    너는 한국 IT 기업에서 실제로 열리는 프로젝트 회의를 작성하는 전문 작가다.
    회의 주제 선택은 네가 직접한다.

    [요구사항]
    1) 아래 형식으로 회의 스크립트를 작성한다.
        - 형식: 이름: 대사
        - 한 줄에 한 사람의 발화만
        - 줄바꿈으로 구분
        - 메타설명, 따옴표, 리스트 사용 금지

    2) 회의 길이 느낌: 약 {duration_min}분 분량
       (약 25~40개의 발화 정도)

    3) 등장인물 (랜덤)
        - 5~10인으로 너가 랜덤으로 선택해서 구성해.

    4) 회의 구성
        - 본론: 새로 생성한 회의 주제를 중심으로 깊이 있는 기술·운영 논의
        - 중간중간 잡담 추가 (컨디션, 점심, 일정, 배포 피곤함 등)
        - 마지막: DRI와 기한을 명시한 액션 아이템 정리

    [출력 형식 예시]
    박지은: 모두 들어왔죠? 오늘 일정 괜찮나요?
    김현우: 네, 어제 배포가 늦어져서 조금 피곤하긴 하네요.
    ...

    위 형식을 반드시 지켜라.
    """
```

2.4. TTA 정보통신 용어사전 웹 크롤링

- 출처: TTA 정보통신용어사전 (한국정보통신기술협회)
  - 전 초성 (ㄱ ~ ㅎ) 대상 정보통신용어사전 / 시사 상식 / TTA 표준 / 기타 참고 필드
- 수집 방법: 크롤링 코드 활용 페이지 단위 HTML 파싱으로 용어의 *word_seq*을 추출
- 수집 목적: 실제 IT 분야에서 활용하는 용어를 통한 모델 파인튜닝을 통해 확실한 태스크 추출 위함
- 특징:
  1. IT 도메인 특화 텍스트 데이터
  2. TTA 정보통신용어사전 내 전 초성(ㄱ~ㅎ) 대상 정보통신용어사전, 시사상식, TTA 표준, 기타참고 필드 수집
- 수집 전략:
  1. 페이지 단위 HTML 파싱으로 용어의 *word_seq*을 추출
  2. 내부 AJAX POST 호출로 용어명과 정의를 수집
- 수집량: 41,967개 용어-정의 쌍


```
{"question": "라다 방식이란 무엇인가", "answer": "몇 개의 주파수와 그 주파수의 펄스 시간 위치를 잘 맞추어 정한 상대국만이 교신할 수 있는 무선 통신 방식의 한 가지. 비밀도가 높고, 주파수의 이용도가 좋다."}
```


- 파일명 예시: tta_results_종류_기준자음.*json*
  (예시: tta_resulta_TTA표준_ㄱ.*json)*
- 수집 코드(Python):
  1) 용어-정의 쌍 크롤링



```python
        try:
            html = fetch_first_list(category=category, initial_consonant=initial, page=page, list_count=list_count)
            values = extract_first_seq_values(html)
            print(f"  - p{page}: {values}")

            for value in values:
                try:
                    data = fetch_detail(value)  # kor_subject / contents(정제) 반환
                    print(f"    {data['word_seq']} | {data['kor_subject']}")
                    print(f"    {data['contents']}")
                    results.append({data['kor_subject']: data['contents']})
```

2.5. 용어집 **PDF** 텍스트 추출

- 출처: 최신ICT시사용어2025 (한국정보통신기술협회)
- 수집 방법: pdf 추출 코드 활용 pdf 내 용어-정의 텍스트 추출
- 수집 목적: 실제 IT 분야에서 활용하는 용어를 통한 모델 파인튜닝을 통해 확실한 태스크 추출 위함
- 특징:
    1. 한국정보통신기술협회에서 발간한 '최신ICT시사용어2025' 에서 추출한 IT 도메인 특화 텍스트 데이터
    2. 용어-정의 쌍 형태로 구성
- 수집량: 79개 용어-정의 쌍


```
{"question": "검색 증강 생성란 무엇인가", "answer": "대규모 언어 모델(LLM)에 쌓인 데이터와 별개의 외부 데이터를 이용해 답변 정확도를 높여주는 기술"}
```


- 파일명 : ict_terms.json
- 수집 코드(Python):

```python
def extract_qa_from_pdf(pdf_path: str):
    qa_list = []

    with pdfplumber.open(pdf_path) as pdf:
        for page_idx, page in enumerate(pdf.pages, start=1):
            text = page.extract_text()
            if not text:
                continue

            lines = text.splitlines()
            term, definition = extract_term_and_def(lines)

            if term and definition:
                question = f"{term}란 무엇인가?"
                qa_list.append({
                    "page": page_idx,
                    "question": question,
                    "answer": definition
                })
                print(f"[{page_idx}쪽] {term} → 추출 완료")

    return qa_list
```

2.6. OpenAI 기반 판교어 단어-정의 쌍 수집

- 수집 방법: OpenAI GPT 모델을 활용하여 판교어 단어-정의 쌍 생성
- 수집 목적: IT 기업 내부에서 쓰는 용어를 지칭하는 말인 '판교어'를 모델에 파인 튜닝하여 회의 중 정확한 IT 용어가 아닌 줄임말, 은어를 사용하더라도 정확한 태스크 추출 위함
- 특징:
    1. OpenAI GPT 모델을 활용하여 생성한 판교어 용어-정의 쌍 형태로 구성
    2. 정의 안에 예시를 추가하여 실제 사용 시 형태를 명시함
    3. OpenAI GPT와 Gemini API 교차 검증을 통한 할루시네이션 방지
- 수집량: 744개 용어-정의 쌍


```
{"question": "탭핑이란 무엇인가", "answer": "'간 본다'는 의미로, 본격 진행 전 유관부서의 반응을 살피는 행위. 가볍게 의견을 묻거나 확인한다는 의미이다. 예: '개발팀에 먼저 탭핑해볼게요.'"}
```


- 파일명 : pangyo_terms.json
- 수집 코드(Python):

```
SYSTEM_PROMPT = """
당신은 IT 기업에서 자주 사용하는 실제 신조어를 생성하는 전문 AI입니다.

반드시 아래 형식으로만 출력하세요:

{"question": "<용어>란 무엇인가?", "answer": "<정의>. 예: '<예문>'"}

규칙:
- 용어는 1~2단어, 한국 실무에서 실제로 사용하는 신조어/판교어/업무 은어
- 정의는 한국어로 작성
- 예문도 한국어로 실제 상황처럼 작성
- JSON 이외의 문장은 절대 출력하지 말 것
- 각 줄은 하나의 JSON만 포함해야 함
"""

USER_PROMPT = """
IT 회사 실무에서 사용될 수 있는 용어 20개를 생성하시오.
각 항목은 반드시 JSON 한 줄로 출력하시오.
"""
```

2.6.1 생성된 판교어 단어-정의 쌍 교차검증 (Gemini 기반)

- 목적: 단일 모델(OpenAI GPT)이 생성한 정의에 대해 사실성 오류·개념 왜곡·누락 여부를 판별하기 위해 서로 다른 LLM(OpenAI GPT, Google Gemini)를 이용해 상호 검증하며 데이터 신뢰도를 보증함.
- 검증 필요성:
    1. 기업 내부 용어(판교어)는 일반 사전에서 찾기 어렵고, 동일한 의미라도 현업에서 쓰는 뉘앙스와 정의가 다름.
    2. 단일 모델 출력만 사용할 경우, 할루시네이션, 일반적 용어와의 혼동, 논리적 비약 등의 위험이 존재

- 교차검증 코드(Python):

```python
def load_pangyo_terms(path: str = "pangyo_terms.json"):
    with open(path, "r", encoding="utf-8") as f:
        return json.load(f)

def verify_term(term_data: dict) -> str:
    prompt = f"""
너는 한국 IT 회사에서 사용하는 신조어/판교어 정의를 검수하는 리뷰어다.

아래 JSON의 question, answer를 보고 정의의 품질을 평가하라.

[입력 JSON]
{json.dumps(term_data, ensure_ascii=False)}

검증 기준:
1) question(용어)와 answer(정의)가 서로 잘 맞는지
2) 정의 내용에 사실성 오류나 이상한 개념이 없는지
3) 실제 한국 IT 실무에서 자연스럽게 쓸 수 있는 표현인지
4) 의미 전달에 중요한 요소가 빠져 있지는 않은지
5) 전반적인 품질에 대해 1~5점으로 점수화

아래 JSON 형식으로만 답변하라:
{{
  "question": "{term_data['question']}",
  "definition_review": "",
  "factual_or_concept_issues": "",
  "missing_key_elements": "",
  "naturalness_in_it_context": "",
  "overall_score": ""
}}
"""
    response = model.generate_content(prompt)
    return response.text
```

2.7. 판교어 용어집 **PDF** 텍스트 추출

- 출처: 사회초년생 및 IT 입문자를 위한 판교어 300선 (크몽)
- 수집 방법: pdf 추출 코드 활용 pdf 내 용어-정의 텍스트 추출
- 수집 목적: IT 기업 내부에서 쓰는 용어를 지칭하는 말인 '판교어'를 모델에 파인 튜닝하여 회의 중 정확한 IT 용어가 아닌 줄임말, 은어를 사용하더라도 정확한 태스크 추출 위함
- 특징:
    1. 파인튜닝 데이터 증강을 위해 추가 수집
    2. 용어-정의 쌍 형태로 구성
- 수집량: 300개 용어-정의 쌍


```
{"question": "카나리 배포(Canary Deployment)(이)란 무엇인가?", "answer": "새로운 버전을 일부 사용자에게만 배포하여 안정성을 확인하는 방식 예: '이번 업데이트는 카나리 배포로 진행해서 문제 여부를 먼저 확인할 계획이야.'"}
```


- 파일명 : finetune_glossary.jsonl
- 수집 코드(Python):

```python
def extract_entries_from_tables(pdf_path, start_page=6):
    entries = []

    with pdfplumber.open(pdf_path) as pdf:
        for i in range(start_page, len(pdf.pages)):
            page = pdf.pages[i]
            tables = page.extract_tables()
            if not tables:
                continue

            for table in tables:
                for row in table:
                    if not row:
                        continue

                    row = list(row) + [""] * (4 - len(row))
                    num, term, meaning, example = row[:4]

                    if isinstance(num, str) and "번" in num and "호" in num:
                        continue
                    if not num or not term:
                        continue

                    entries.append({
                        "num": clean_cell(num),
                        "term": clean_cell(term),
                        "meaning": clean_cell(meaning),
                        "example": clean_cell(example),
                    })

    return entries
```

```python
def entries_to_jsonl(entries, output_path):
    with open(output_path, "w", encoding="utf-8") as f:
        for e in entries:
            term = e["term"]

            meaning_raw = e["meaning"]
            example_raw = e["example"]

            meaning = fix_spacing(meaning_raw)
            example = fix_spacing(example_raw)

            question = f"{term}(이)란 무엇인가?"

            answer = f"{meaning} 예: {example}"

            obj = {
                "question": question,
                "answer": answer,
            }
            f.write(json.dumps(obj, ensure_ascii=False) + "\n")


if __name__ == "__main__":
    entries = extract_entries_from_tables(PDF_PATH, START_PAGE)
    entries_to_jsonl(entries, OUTPUT_PATH)
    print(f"JSONL 파일 생성 완료: {OUTPUT_PATH}, 총 {len(entries)}개 항목")
```

```
JSONL 파일 생성 완료: pangyo_dict.jsonl, 총 300개 항목
```