
수집 데이터 및 데이터 전처리 보고서

2025년 11월 17일

1. 문서 개요

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

본 프로젝트는 음성 인식 모델인 **Whisper**와 화자분리 모델 **Pyannote**를 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	RAG용 도메인 특화 용어집 (IT)	(한국정보통신기술협회)	41,967개	18.46 MB	IT 용어 크롤링
텍스트	RAG용 도메인 특화 용어집 (IT)	(한국정보통신기술협회)	79개		IT 용어 pdf 추출
텍스트	RAG용 도메인 특화 용어집 (IT)	OpenAI GPT 모델 생성	744개		판교어 정의쌍
텍스트	RAG용 도메인 특화 용어집 (IT)	크몽 - 사회초년생 및 IT 입문자를 위한 판교어 300선	300개		판교어 pdf 추출
텍스트	RAG용 도메인 특화 용어집 (회계)	국립국어원 용어 검색 서비스	520개	1.4 MB	회계 용어 크롤링
텍스트	RAG용 도메인 특화 용어집 (회계)	국세청 세법 용어집	1,728개		
텍스트	RAG용 도메인 특화 용어집 (디자인)	Openpath 디자인 용어집	331개	3.3 MB	디자인 용어 크롤링
텍스트	RAG용 도메인 특화 용어집 (디자인)	한국문화예술위원회 시각예술 용어	300개		
텍스트	RAG용 도메인 특화 용어집 (디자인)	국립국어원 용어 검색 서비스	12,634개		
텍스트	RAG용 도메인 특화 용어집 (마케팅/경제)	DFINERY 마케팅 용어집	92개	11 MB	마케팅 용어 크롤링
텍스트	RAG용 도메인 특화 용어집 (마케팅/경제)	Adriel 마케팅 사전	45개		
텍스트	RAG용 도메인 특화 용어집 (마케팅/경제)	Ascentkorea 용어집	80개		경제·마케팅 용어 크롤링
텍스트	RAG용 도메인 특화 용어집 (마케팅/경제)	한국경제 경제용어사전	13,137개		경제 용어 크롤링

텍스트	파인튜닝용 합성 회의 데이터셋(전문·안건·태스크 통합)	OpenAI GPT 모델 (합성 데이터)	2,621개	7.4 MB	회의 전문·안건·태스 크 통합 생성 데이터
-----	--------------------------------------	---------------------------	--------	--------	----------------------------------

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

```
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

```

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 팀 회의

- 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 기업에서 진행하는 회의 상황을 반영하기 위해 고품질 회의 스크립트 추가 생성

하여 **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 내가 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
        - 본문: 새로 생성한 회의 주제를 중심으로 깊이 있는 기술·운영 논의
        - 중간중간 잠깐 추가 (컨디션, 점심, 일장, 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 IT 도메인 특화 용어집

2.4.1 TTA 정보통신 용어사전 웹 크롤링

- 출처: TTA 정보통신용어사전 (한국정보통신기술협회)
 - 전 초성 (ㄱ ~ ㅎ) 대상 정보통신용어사전 / 시사 상식 / TTA 표준 / 기타 참고 필드
- 수집 방법: 페이지 단위 HTML 파싱을 통해 용어의 **word_seq**을 추출
- 수집 목적: IT 도메인에 특화된 **RAG** 지식베이스 구축을 위해 표준화된 용어·정의 데이터를 확보하여, 검색 정확도 및 문맥 기반 응답 품질을 향상하기 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  
상대국  
만이 교신할 수 있는 무선 통신 방식의 한 가지. 비밀도가 높고,  
주파수의  
이용도가 좋다." }
```

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

```
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']})  
        except:
```

2.4.2 용어집 PDF 텍스트 추출

- 출처: [최신 ICT 시사용어 2025](#) (한국정보통신기술협회)
- 수집 방법: pdf 추출 코드를 활용하여 문서 내 용어-정의 텍스트를 구조화된 형태로 추출
- 수집 목적: IT 도메인 특화 RAG 지식베이스 구축을 위해 표준화된 용어-정의 데이터를 확보하여,

검색 및 문맥 기반 응답 품질을 향상하기 위함

- 특징:

1. 한국정보통신기술협회에서 발간한 '최신 ICT 시사용어 2025' 문서 기반 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):

```
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
```


2.4.3 OpenAI 기반 판교어 단어-정의 쌍 수집

- 수집 방법: OpenAI GPT 모델을 활용하여 판교어 단어-정의 쌍 생성하고, 품질 검증을 위해

Gemini API와 교차 비교 수행

- 수집 목적: IT 기업 내부에서 사용되는 비표준 용어(판교어)를 RAG 지식베이스에 포함하여,

회의·대화 등에서 비정형 표현이나 줄임말이 등장하더라도 정확한 의미 해석 및 문맥

기반 태스크 추출이 가능하도록 하기 위함

- 특징:

1. OpenAI GPT 모델 기반 생성 데이터로 구성된 판교어 용어-정의 쌍
2. 실제 사용 맥락을 반영하기 위해 각 정의에 예시 문장을 포함
3. OpenAI GPT 및 Gemini API 간 교차 검증을 통해 생성 품질 및 할루시네이션 위험 최소화

- 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 생성하십시오.  
각 항목은 반드시 JSON 한 줄로 출력하십시오.  
"""
```

- 생성된 판교어 단어-정의 쌍 교차검증 (Gemini 기반):

단일 모델(OpenAI GPT)이 생성한 정의의 경우, 할루시네이션, 일반적 용어와의 혼동, 논리적

비약 등의 위험이 존재. 사실성 오류·개념 왜곡·누락 여부를 판별하기 위해 서로 다른 LLM

(OpenAI GPT, Google Gemini)를 이용해 상호 검증하며 데이터 신뢰도를 보증함.

```
11 def verify_term(term_data: dict) -> str:
12     prompt = f"""
13     너는 한국 IT 회사에서 사용하는 신조어/판교어 정의를 검수하는 리뷰어다.
14
15     아래 JSON의 question, answer를 보고 정의의 품질을 평가하라.
16
17     [입력 JSON]
18     {json.dumps(term_data, ensure_ascii=False)}
19
20     검증 기준:
21     1) question(용어)와 answer(정의)가 서로 잘 맞는지
22     2) 정의 내용에 사실성 오류나 이상한 개념이 없는지
23     3) 실제 한국 IT 실무에서 자연스럽게 쓸 수 있는 표현인지
24     4) 의미 전달에 중요한 요소가 빠져 있지는 않은지
25     5) 전반적인 품질에 대해 1~5점으로 점수화
```

2.4.5 판교어 용어집 PDF 텍스트 추출

- 출처: [사회초년생 및 IT 입문자를 위한 판교어 300선](#) (크몽)
- 수집 방법: pdf 추출 코드를 활용하여 문서 내 용어-정의 텍스트를 구조화된 형태로 수집
- 수집 목적: IT 기업 내부에서 사용되는 비표준 용어(판교어)를 RAG 지식베이스에 포함하여,
회의·실무 커뮤니케이션에서 등장하는 줄임말·은어·관용 표현에 대한 문맥 해석
및
태스크 추출 정확도를 향상하기 위함
- 특징:
 1. 크몽에서 제공하는 판교어 PDF 자료 기반의 용어-정의 텍스트 데이터
 2. RAG 검색 및 의미 기반 매칭에 적합하도록 단어-정의 쌍 형태로 구성
 3. 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):

```
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
```

```
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")
```

2.5. 회계 특화 용어집 텍스트 추출

2.5.1 국립국어원 회계 용어 텍스트 추출

- 출처: [국립국어원 용어 검색 서비스](#)

- 수집 방법: 웹 페이지 단위 **HTML** 파싱을 통해 회계 관련 용어와 정의 텍스트를 추출

- 수집 목적: 회계 도메인 특화 **RAG** 지식베이스를 구축하고 의미 기반 검색 품질을 향상하기 위함

- 특징:

1. 국립국어원에서 제공하는 공식 용어·정의 기반 회계 도메인 특화 텍스트 데이터
2. 표준화된 정의 형식으로 수집되어 **RAG** 인덱싱 및 검색에 적합
3. 용어-정의 쌍 구조로 구성되어 의미 매칭 품질 향상에 효과적

- 수집량: 520개 용어-정의 쌍

```
{ "title": "가속상각",  
  "answer": "고정자산에 투자된 금액을 조기에 회수하기 위하여 특정고정자산에  
대한  
감가상각을 통상의 감가상각보다 짧은 기간에 행하는 것을  
말한다.",  
  "domain": "Accounting",  
  "dept_id": "004" }
```

- 파일명 : rag_accounting.json

- 수집 코드(Python):

```

MAIN_URL = "https://kli.korean.go.kr/term/indexMain.do?lang=kr"

DOMAIN = "Accounting"
DEPT_ID = "004"
OUTPUT_JSON = "kli_terms_426_3_all.json"

TOTAL_PAGES = 50
BLOCK_SIZE = 5

def create_driver():
    options = webdriver.ChromeOptions()
    options.add_argument("--no-sandbox")
    options.add_argument("--disable-dev-shm-usage")
    driver = webdriver.Chrome(
        service=Service(ChromeDriverManager().install()),
        options=options
    )
    return driver

def open_search_result_list(driver):
    driver.get(MAIN_URL)
    time.sleep(2)

    # 1. 대분류 클릭
    box = driver.find_element(By.ID, "lclas_name_425")
    box.find_element(By.TAG_NAME, "p").click()
    time.sleep(1)

    # 2. 버블 3번째 클릭
    driver.find_element(
        By.XPATH, '//*[@id="bubble_lst_425"]/li[6]/a'
    ).click()

    # 3. 리스트 페이지 로딩 대기
    WebDriverWait(driver, 10).until(
        EC.url_contains("indexSearchList.do")
    )
    WebDriverWait(driver, 10).until(
        EC.presence_of_element_located((By.ID, "includeSearchList"))
    )

```

2.5.2 국세청 세법 용어집 텍스트 추출

- 출처: [국세청 세법 용어집](#)

- 수집 방법: 문서 내 용어 페이지를 크롤링하여 용어명 및 정의를 구조화된 텍스트로 수집.
세법 관련 용어 항목을 필터링하여 회계·세무 도메인 중심 데이터셋 구축

- 수집 목적: 회계 도메인 특화 **RAG** 지식베이스를 구축하고 의미 기반 검색 품질을 향상하기 위함

- 특징:

1. 국세청에서 제공하는 공식 세법 용어 정의 기반 데이터
2. 회계·세무 도메인의 실제 업무 문맥에 특화된 용어·정의 쌍
3. 전문 용어가 많아 **RAG** 기반 추론의 정확성 향상에 기여

- 수집량: 1,728개 용어-정의 쌍

```

{ "title": "가산세",
  "answer": "세법에서 규정하는 의무의 성실한 이행을 확보하기 위하여 세법에 따라 산출한 세액에 가산하여 징수하는 금액을 말한다(국기법 24호, 지기법 52). 가산세는 협력의무 위반에 대한 행정벌적 성격을 그 본질로 한다고 보는 것이 통설이다. 그러나 세금의 형식으로 부과되기 때문에 조세벌과는 구별되며, 동일한 의무위반에 대하여 가산세와 조세벌이 중복적으로 과징되어도 이는 이중처벌이 아니라고 본다. 요컨대 가산세는 각 가산세의 부과를 규정하고 있는 세법에 의해

```

징수되는 세금에 부가하여 징수되기 때문이다. [참조조문] 국기법 24호, 법률
75~75의 9, 소법 81, 부법 60",
"domain": "Accounting",
"dept_id": "004" }

- 파일명 : rag_accounting.json

- 수집 코드(Python):

```
BASE_LIST_URL = "https://taxlaw.nts.go.kr/st/USESTJ001P.do"

# 결과 저장 리스트
results = []

def create_driver():
    options = webdriver.ChromeOptions()
    options.add_argument("--no-sandbox")
    options.add_argument("--disable-dev-shm-usage")

    driver = webdriver.Chrome(options=options)
    driver.maximize_window()
    return driver

def wait_and_get_answer(driver, wait, timeout=10):
    try:
        wait.until(
            EC.presence_of_element_located((By.ID, "ntstTrgyDictEpnCntn"))
        )
        def text_not_empty(d):
            try:
                el = d.find_element(By.ID, "ntstTrgyDictEpnCntn")
                txt = el.text.strip()
                if txt:
                    return True
                inner = el.get_attribute("innerText") or ""
                return inner.strip() != ""
            except Exception:
                return False

        WebDriverWait(driver, timeout).until(text_not_empty)

        el = driver.find_element(By.ID, "ntstTrgyDictEpnCntn")
        txt = el.text.strip()
        if not txt:
            txt = (el.get_attribute("innerText") or "").strip()
        return txt
```

2.6. 디자인 특화 용어집 텍스트 추출

2.6.1 Openpath 디자인 용어집 크롤링

- 출처: [Openpath 디자인 용어집](#) (디자이너를 위한 교육 콘텐츠 플랫폼)
- 수집 방법: 웹 페이지 단위 HTML 파싱을 통해 디자인 용어 및 정의를 추출
- 수집 목적: 디자인 도메인 특화 RAG 지식베이스를 구축하고 의미 기반 검색 품질을 향상하기 위함
- 특징:
 1. 다양한 디자인 영역을 포괄하는 실무 중심 용어·정의 데이터
 2. 용어·정의 쌍 형태로 구조화되어 의미 매칭 및 검색 성능 향상에 적합
- 수집량: 331개 용어-정의 쌍

```
{ "title": "San-Serif (산세리프)",  
  "answer": "글자와 기호를 이루는 획의 일부 끝이 돌출되는 세리프가 없는  
형태.",  
  "domain": "Design",  
  "dept_id": "003" }
```

- 파일명 : rag_design.json

- 수집 코드(Python):

```
BASE_URL = "https://openpath.kr/knowhow/open-wiki?category=&text=&page={page}"  
  
DOMAIN = "Design"  
DEPT_ID = "003"  
  
headers = {  
    "User-Agent": "OpenpathGlossaryCrawler/1.0 (contact: your-email@example.com)"  
}  
  
def fetch_page_html(page: int) -> str:  
    url = BASE_URL.format(page=page)  
    resp = requests.get(url, headers=headers, timeout=10)  
    resp.raise_for_status()  
    return resp.text  
  
def parse_terms(html: str):  
    soup = BeautifulSoup(html, "html.parser")  
    results = []  
  
    # #contents 안의 li 들 중에서, h4 > span > span 과 p 가 같이 있는 것만 사용  
    for li in soup.select("#contents li"):  
        title_span = li.select_one("h4 span span")  
        p_tag = li.select_one("p")  
        if not title_span or not p_tag:  
            continue  
  
        title = title_span.get_text(strip=True)  
        answer = p_tag.get_text(" ", strip=True)  
  
        results.append({  
            "title": title,  
            "answer": answer,  
            "domain": DOMAIN,  
            "dept_id": DEPT_ID  
        })  
    return results
```

2.6.2 한국문화예술위원회 시각예술 용어집 추출

- 출처: [한국문화예술위원회 시각예술 용어](#)

- 수집 방법: 페이지별 HTML 파싱을 통해 시각예술 관련 용어 및 정의 수집. 용어 카테고리 기반

필터링을 통해 디자인 분야에 해당하는 항목만 추출

- 수집 목적: 예술·디자인 도메인 특화 **RAG** 지식베이스를 구축하고 의미 기반 검색 품질을 향상하기

위함

- 특징:

1. 예술 및 디자인 학문 기반 공식 용어·정의 데이터
2. 시각예술 중심 용어로 구성되어 비주얼·조형 관련 개념 해석에 유용
3. **RAG** 의미 기반 검색 및 문맥 이해 정확도 향상에 기여

- 수집량: 300개 용어-정의 쌍

```
{ "title": "모노크롬",  
  "answer": "단일한 색조에 명도와 채도의 변화만 있는 추상 회화의 한 종류이다.  
             매체의 순수성, 평면성, 환원성을 지향한 1960~1970년대 미국의  
             모더니스트 회화를 지칭하기도 한다.",  
  "domain": "Design",  
  "dept_id": "003" }
```

- 파일명 : rag_design.json

- 수집 코드(Python):


```

BASE_LIST_URL = "https://www.gokams.or.kr/visual-art/art-terms/glossary/art_list.asp"
BASE_DETAIL_URL = "https://www.gokams.or.kr/visual-art/art-terms/glossary/art_view.asp?idx={idx}&page=1&s1=&s2=&flag_initial="

DOMAIN = "Design"
DEPT_ID = "003"

OUTPUT_FILE = "art_terms.jsonl"

def create_driver():
    options = webdriver.ChromeOptions()
    options.add_argument("--no-sandbox")
    options.add_argument("--disable-dev-shm-usage")

    return webdriver.Chrome(
        service=Service(ChromeDriverManager().install()),
        options=options
    )

def click_more_until_end(driver):
    while True:
        try:
            btn = driver.find_element(By.XPATH, '//*[@id="btn_more"]')
            if btn.is_displayed():
                btn.click()
                time.sleep(1)
            else:
                break
        except:
            break

def extract_list_items(driver):
    items = driver.find_elements(By.XPATH, '//*[@id="html_data_list"]/tr/td[1]/a')
    results = []

    for a in items:
        title = a.text.strip()
        href = a.get_attribute("href")

        match = re.search(r"go_view\('(\\d+)'\)", href)
        if not match:
            continue

        idx = match.group(1)
        results.append({"title": title, "idx": idx})

    return results

```

2.6.3 국립국어원 디자인 용어 텍스트 추출

- 출처: [국립국어원 용어 검색 서비스](#)
- 수집 방법: HTML 파싱으로 용어명·정의 텍스트를 구조화하여 용어-정의 쌍으로 변환
- 수집 목적: 디자인·예술 도메인 특화 **RAG** 지식베이스를 구축하고 의미 기반 검색 품질 향상하기
위함
- 특징:
 1. 다양한 디자인 영역을 포괄하는 실무 중심 용어·정의 데이터
 2. 용어-정의 쌍 형태로 구조화되어 의미 매칭 및 검색 성능 향상에 적합
- 수집량: 12,634개 용어-정의 쌍

```
{ "title": "가변식가구",  
  "answer": "구조를 바꾸면 다른 용도로 쓸 수 있도록 만든 가구.",  
  "domain": "Design",  
  "dept_id": "003" }
```

- 파일명 : rag_design.json
- 수집 코드(Python): 본 데이터는 2.5.1과 동일한 Python 기반 HTML 파싱 코드 구조를 사용하였으며, 대상 URL만 다름.

2.7. 마케팅/경제 특화 용어집 텍스트 추출

2.7.1 DFINERY 마케팅 용어집 크롤링

- 출처: [DFINERY 마케팅 용어집](#)

- 수집 방법: 웹 페이지 HTML 파싱을 활용하여 마케팅 관련 용어 및 정의 텍스트 추출. 단일 페이지

및 다중 목록 구조를 분석하여 용어-정의 쌍으로 구조화

- 수집 목적: 디지털 마케팅·퍼포먼스 마케팅 분야의 주요 개념을 RAG 지식베이스에 반영하여

마케팅 관련 용어 및 지표에 대한 정확한 의미 기반 검색과 문맥 해석을 지원하기 위함

- 특징:

1. 실무 중심 마케팅 분석 용어 기반 텍스트 데이터
2. 용어-정의 쌍 형태로 정규화되어 RAG 검색 품질 향상에 기여

- 수집량: 92개 용어-정의 쌍

```
{ "title": "STP",  
  "answer": "시장 세분화(Segmentation), 표적 타겟팅(Targeting), 차별적  
            포지셔닝(Positioning)을 통해 수익성을 극대화하는 전략.",  
  "domain": "Marketing / Economy",  
  "dept_id": "002" }
```

- 파일명 : rag_marketing_economy.json

- 수집 코드(Python): 본 데이터는 2.6.1과 동일한 Python 기반 HTML 파싱 코드 구조를 사용하였으며, 대상 URL만 다름.

2.7.2 Adriel 마케팅 용어집 크롤링

- 출처: [Adriel 마케팅 사전](#)

- 수집 방법: 웹 페이지 HTML 파싱을 활용하여 용어 및 정의 텍스트 추출. 특정 필드(마케팅 전략·운영 지표 등)를 중심으로 텍스트 데이터 구조화

- 수집 목적: 디지털 광고 운영에서 빈번하게 사용되는 전문 용어를 RAG 지식베이스에 반영하여

마케팅 관련 용어 및 지표에 대한 정확한 의미 기반 검색과 문맥 해석을 지원하기 위함

- 특징:

1. 광고·운영·성과 분석 관련 용어 중심 데이터
2. 간결한 정의로 구성되어 용어 매칭 및 요약 질의 처리에 효과적

- 수집량: 45개 용어-정의 쌍

```
{ "title": "CPC",  
  "answer": "제품개발 환경에서 기업 간, 부서 간 동시작업과 협력을 가능케 하는  
            컴퓨터를 이용한 경영기법.",  
  "domain": "Marketing / Economy",  
  "dept_id": "002" }
```

- 파일명 : rag_marketing_economy.json

- 수집 코드(Python): 본 데이터는 2.6.1과 동일한 Python 기반 HTML 파싱 코드 구조를 사용하였으며, 대상 URL만 다름.

2.7.3 Ascentkorea 용어집 텍스트 추출

- 출처: [Ascentkorea 용어집](#)
- 수집 방법: 웹 페이지 기반 **HTML** 파싱으로 용어 및 정의를 수집
- 수집 목적: 경제·마케팅 도메인 특화 **RAG** 지식베이스를 구축하고 의미 기반 검색 품질을 향상하기 위함
- 특징:
 1. 마케팅 전략, 스타트업 운영, 비즈니스 개념 등 실무 기반 용어 포함
 2. 정의가 비교적 서술형으로 제공되어 문맥 기반 응답 품질 향상에 기여
- 수집량: 80개 용어-정의 쌍

```
{ "title": "Backlink",  
  "answer": "외부 사이트에서 특정 웹사이트로 연결되는 링크를 의미한다. 검색  
엔진  
최적화(SEO)에서 중요한 신뢰도 지표로 사용된다.",  
  "domain": "Marketing / Economy",  
  "dept_id": "002" }
```

- 파일명 : rag_marketing_economy.json
- 수집 코드(Python): 본 데이터는 2.6.1과 동일한 Python 기반 HTML 파싱 코드 구조를 사용하였으며, 대상 URL만 다름.

2.7.4 한국경제 경제용어사전 크롤링

- 출처: [한국경제 경제용어사전](#)

- 수집 방법: 페이지 크롤링을 수행하여 경제 용어 및 정의 텍스트 수집. 다량의 용어를 구조화하기

위해 용어-정의 필드를 정규화하여 저장

- 수집 목적: 경제·금융·시장 분석에 활용되는 전문 용어를 폭넓게 확보하여 경제 도메인 RAG

지식베이스의 개념 커버리지 및 문맥 해석 능력을 강화하기 위함

- 특징:

1. 경제 정의 특성상 금융지표·경제학 개념 등 복합 텍스트 구조 포함
2. RAG 임베딩 및 검색 성능 향상에 중요한 역할 수행

- 수집량: 13,137개 용어-정의 쌍

```
{ "title": "가격우선의 원칙",  
  "answer": "증권시장의 경쟁매매에 있어 호가의 우선순위를 정하는 것으로서 파는  
경우는 낮은 가격을, 사는 경우는 높은 가격을 우선한다는 말이다. 가장 불리한  
주문, 예를 들어 가장 낮은 가격의 매도 주문이나 가장 높은 가격의 매입 주문에  
대하여 우선적으로 거래를 성립시키는 방법이다. 시간우선의 원칙, 수량우선의 원칙과  
함께 쓰인다.",  
  "domain": "Marketing / Economy",  
  "dept_id": "002" }
```

- 파일명 : rag_marketing_economy.json

- 수집 코드(Python):

```

BASE_URL = "https://dic.hankyung.com/economy/list/?phoneme={phoneme}"

DOMAIN = "Marketing / Economy"
DEPT_ID = "002"

def create_driver():
    options = webdriver.ChromeOptions()
    options.add_argument("--headless")
    options.add_argument("--no-sandbox")
    options.add_argument("--disable-dev-shm-usage")
    driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()),
                              options=options)
    return driver

def click_all_more_terms(driver):
    while True:
        try:
            btn = driver.find_element(By.ID, "moreBtn")

            if not btn.is_displayed() or not btn.is_enabled():
                print("더보기 버튼 비활성 → 종료")
                break

            driver.execute_script("arguments[0].click();", btn)
            time.sleep(1.0)

        except NoSuchElementException:
            print("더보기 버튼 없음 → 종료")
            break

```

2.8 파인튜닝용 합성 회의 데이터셋 (전문·안건·태스크 통합)

- 출처: OpenAI GPT 모델 기반 합성 데이터

- 수집 방법:

1. Python 스크립트에서 OpenAI API(gpt-4-turbo)를 호출하여 회의 전문과 분석 결과를 동시에 생성하도록 구성.
2. 공통 시스템 프롬프트를 통해 모델에게 다음 조건을 강제
 - a. 회의 전문은 실제 STT 전사처럼 문장 부호 없이, 구어체·비문·간투사를 포함해 생성
 - b. 화자 구분은 '화자명:' 형식으로만 표기
 - c. 최종 출력 JSON은 반드시 {"transcript":..., "agendas": [...], "tasks": [...]} 구조를 따를 것
 - d. tasks 배열의 각 원소는 who, what, when 필드를 반드시 포함하고, 정보가 없으면 "null"로 기입
3. 미리 작성된 예시 3개를 few-shot 형태로 messages에 포함하여, 회의 전문과 agendas·tasks가 어떤 식으로 매핑되어야 하는지 유도
4. 도메인(IT 개발, 마케팅, 법무/컴플라이언스, 인사/교육, 재무/회계 등)과 역할(PM, 개발자, 디자이너, 팀장, 외부 컨설턴트 등)을 조합하여, 매 호출마다 서로 다른 도메인·역할 구성이 되도록 프롬프트를 생성하고 이를 바탕으로 새로운 회의 시나리오와 JSON 분석 결과를 동시에 생성

- 수집 목적: 회의 전문에서 주요 안건 목록(agendas)과 태스크(tasks)를 추출하는 sLLM 파인튜닝용 학습 데이터를 구축하기 위함. 특히 태스크의 선후 관계, 담당자 변경, 기한

지연 등 현실적인 회의 흐름을 포함한 시나리오를 합성하여, 실제 업무 회의 녹취를

입력하였을 때도 안정적으로 안건 요약과 태스크 추출을 수행할 수 있도록 하는 데

목적이 있음

- 특징:

1. 하나의 레코드에 회의 전문, 안건 리스트, 태스크가 모두 포함된 통합 JSON 구조 데이터셋
2. STT 전사 특성을 반영하여 문장 부호가 없고, 구어체·간투사·비문이 섞인 형태로 전문이 생성되어 실제 음성 인식 후처리 환경과 유사
3. 태스크 담당자와 기한 등이 일부만 언급되거나 변경되는 경우를 포함하도록 시스템 프롬프트에서 명시적으로 요구하여, 다양한 상황을 학습할 수 있게 설계됨
4. 도메인과 역할 조합을 다양화하여 특정 조직 및 산업에 치우치지 않은 범용 회의 시나리오를 포함

5.


```
{ "messages":
  [ { "role": "system",
    "content": "당신은 회의록 전문을 분석하는 AI입니다. 반드시 다음 JSON
스키마로만 출력하세요. agendas는 주요 안건을 3~6개 이내로 요약하고, tasks는
모든 담당자와 할 일, 기한을 빠짐없이 추출하세요."
  }
```

출력 스키마 (다른 말 없이 이 형식만 출력):

```
{
  \\"agendas\\": [\\"안전1\\", \\"안전2\\", ...],
  \\"tasks\\": [{\\"who\\": \\"담당자 이름 또는 역할 (없으면 '미정')\\",
    \\"what\\": \\"구체적인 할 일 내용\\",
    \\"when\\": \\"기한 또는 예상 시점 (명확히 언급되지 않으면
'미정',
                                     모호하면 원문 표현 그대로 기록 예: '다음 회의
전',
                                     '내일까지')\\"
    ]}],
  \\"role\\": \\"user\\",
  \\"content\\": \\"다음 회의록에서 주요 안전과 누가 언제까지 무엇을 하기로 했는지
JSON 형식으로 추출하세요.\\"
}
```

회의록: 민수: 오늘 회의 주제는 새로운 프로젝트의 개발 일정과 작업 분배입니다
먼저 현재 UI 디자인 진행 상황부터 들어보겠습니다\n지훈: 디자인 팀에서는 메인
페이지 레이아웃을 거의 완성했습니다 다만 상세 페이지는 아직 초안 단계에
있어요\n민수: 알겠습니다 그럼 상세 페이지 디자인이 끝나야 개발 시작할 수
있겠네요 현주 님, 상세 페이지 디자인 언제쯤 완료될까요\n현주: 다음 주
금요일까지는 마무리 지을 수 있을 것 같습니다\n민수: 그럼 개발은 상세 페이지
디자인 완료 후에 시작하는 걸로 하죠 지훈 님, 그 사이에 메인 페이지 코드 리뷰를
부탁드립니다\n지훈: 네, 리뷰는 이번 주 중으로 끝내겠습니다\n은지: QA 관련해서는
리뷰가 끝나면 바로 테스트 준비를 시작하겠습니다 그리고 버그 리포트도
준비할게요\n민수: 좋습니다 그럼 QA는 지훈 님 코드 리뷰가 끝나는 대로 진행하시고
현주 님이 디자인을 마친 후에 테스트 포함된 전체 일정을 다시 조율하도록
하겠습니다"}.

```
{
  "role": "assistant",
  "content": "{
    \"agendas\": [\"UI 디자인의 현재 진행 상황 및 완료 예정일 확인\",
      \"개발 일정 조정 및 코드 리뷰\",
      \"QA 준비와 테스트 일정 조율\"],
    \"tasks\": [{\"who\": \"현주\",
      \"what\": \"상세 페이지 디자인 완성\",
      \"when\": \"다음 주 금요일\"},
      {\"who\": \"지훈\",
      \"what\": \"메인 페이지 코드 리뷰\",
      \"when\": \"이번 주 중\"},
      {\"who\": \"은지\",
      \"what\": \"코드 리뷰 완료 후 테스트 및 버그 리포트 준비\",
      \"when\": \"미정\"}
    ]
  }
}
```

```
}}}
```

- 파일명 :syn_data.jsonl

- 수집 코드(Python):

```
108
109 SYSTEM_PROMPT = """
110 당신은 회의 전문을 분석하여 안건과 할 일을 추출하는 고수는 AI 데이터 생성기입니다.
111 당신이 생성할 데이터는 SLLM의 파인튜닝에 사용될 것이므로, **아래의 모든 조건들을 엄격하게 준수**해야 합니다.
112
113 <난이도 및 추출 조건>
114 1. **태스크 종속성 및 재정의:** 논의 도중 태스크의 순서나 책임이 바뀌고, 최종 결정된 워크플로우(선행-후행 관계)를 따라야 합니다.
115 2. **자연어 결정/불완전:** 태스크의 담당자(who), 내용(what), 기한(when) 중 **하나 또는 여러 개가 전문에서 명확히 언급되지 않아** JSON 출력 시 해당 필드가 `null`로 추론
116 3. **암묵적 안건:** 안건 중 하나는 명시적인 제목이 아닌, 대화 내용을 종합하여 추론되는 함축적인 제목이어야 합니다.
117 4. **비선택적 대화:** 하나의 안건을 논의하던 중 다른 안건으로 잠시 **이탈(Off-topic)**했다가 다시 돌아오는 흐름을 반드시 포함해야 합니다.
118
119 <STT 전문 형식성 조건>
120 1. **구어체 및 비문:** 생성하는 전문은 실제 음성을 텍스트로 전사(STT)한 형태이므로, **문장 부호(마침표, 쉼표 등)는 사용하지 마십시오.** 또한, 대화 내용은 비문이나 구어
121 2. **회자 구분:** 회자 명칭은 '화자1:', '팀장:', '민수:'와 같이 **알려진 활동(<) 형태**로만 구분해야 합니다.
122 3. **비언어적 요소 제거:** 비언어적인 행동이나 감정 묘사(예: "(웃음)", "[자료 넘김]")는 **절대 포함하지 마십시오.**
123
124 <출력 형식 제약>
125 1. **다중 추출:** 'agendas'는 **최소 2개 이상**, 'tasks'는 **최소 2개 이상** 추출되어야 합니다.
126 2. **출력 형식:** 출력은 JSON 형식이어야 하며, JSON 객체 외에 어떠한 설명, 서론, 결론도 포함하지 마십시오.
127 3. **JSON 구조:** 최종 출력 JSON은 **'transcript', 'agendas', 'tasks'** 세 가지 키를 포함해야 합니다.
128 4. **태스크 필드:** 'tasks' 객체는 **'who', 'what', 'when'** 필드를 반드시 포함해야 합니다. (단, 정보가 불완전하거나 결정되지 않은 경우 해당 필드의 값은 **'null'***로 채
129 """
130
131 # 데이터 다양성을 위한 시나리오 설정
132 DOMAINS = [
133     "IT 개발팀", "제품 기획팀", "마케팅 전략팀", "법무/컴플라이언스팀",
134     "인사/교육팀", "R&D (연구 개발팀)", "재무/회계팀", "CS/VOC 분석팀",
135     "영업/고객지원팀", "공급망 관리팀"
136 ]
137 ROLES = [
138     ["PM", "개발자", "디자인", "QA 엔지니어"],
139     ["팀장", "과장", "신입사원", "외부 컨설턴트"],
140     ["파트너", "운영팀원", "협력업체 담당자", "CEO"],
141     ["프로젝트 리더", "마케터", "데이터 분석가", "외부 이해관계자"]
142 ]
143
144 MODEL_NAME = "gpt-4-turbo"
145
```

```
155 for transcript, example_json in zip(EXAMPLE_TRANSCRIPTS, EXAMPLE_JSONS):
156     base_messages.append({
157         "role": "user",
158         "content": f"다음 고난도 시나리오에 기반한 회의 전문을 분석해 주세요.\n\n전문:\n{transcript}"
159     })
160     base_messages.append({
161         "role": "assistant",
162         "content": json.dumps(example_json, ensure_ascii=False)
163     })
164
165 generated_count = 0
166
167 with open(output_filename, "a", encoding="utf-8") as f:
168     print(f"💎 {num_to_generate}개의 데이터를 생성합니다. 파일: {output_filename}")
169
170     for i in range(num_to_generate):
171         domain = DOMAINS[i % len(DOMAINS)]
172         roles = ROLES[i % len(ROLES)]
173
174         new_user_prompt = f"""
175         새로운 고난도 회의 전문과 분석 결과를 생성해주세요.
176
177         - 도메인: {domain}
178         - 참가자 역할: {' '.join(roles)}
179         - 요구 난이도: 태스크 종속성 및 자연어 결정이 포함되도록 하세요.
180         - 출력 형식: transcript, agendas, tasks 세 키를 갖는 JSON 객체만 출력하세요.
181         """
```