Data Paper https://doi.org/10.23287/KJDH.2026.3.1.4

중국 상하이도서관 《申報》 텍스트 데이터와 그 수집 및 전처리 절차*

Text data from the Shanghai Library *Shun Pao* database and the workflow for collecting and preprocessing

배건준(Geonjoon, Bae)**
0009-0005-0294-7147

목차

1. 서론
2. 데이터 수집
2.1 수집 스크립트
2.2 수집 결과 데이터
3. 전처리 - 연구용 통합 데이터 제작
4. 결론
부록. 연구 산출물 및 데이터셋 구성

초록

이 논문은 중국 상하이도서관(上海圖書館) 《신보》 데이터베이스 텍스트 데이터 수집 스크립트의 작성 과정과 실제 수집 절차, 수집 데이터의 구조와 특징, 그리고 연구용 통합 데이터 구축을 위한 전처리 스크립트의 처리 절차와 결과물 구조를 설명한다. 스크립트의 설계와 작성 과정 전반에 AI 코딩 에이전트를 활용하여, 사용자가 요구 사항, 오류 메시지 등을 제시하면 이에 따라 AI가 수정과 검증을 반복하는 Human-in-the-Loop 방식으로 작업을 진행하였다. 작업 관련 프롬프트와 코드, 데이터 샘플은 이 연구 GitHub 저장소에 공개한다.

주제어: 《신보》, 상하이도서관, 텍스트 데이터, 디지털 역사학, Human-in-the-Loop

Abstract

This paper presents a workflow for collecting and preprocessing text data from the Shanghai Library *Shun Pao* database. It describes the development of a text-data collection script, the actual collection procedure, the structure and characteristics of the collected data, and the preprocessing procedure and output structure used to construct an integrated research dataset. The scripts were designed and developed with the assistance of an AI coding agent

*이 논문은 2023년 대한민국 교육부와 한국연구재단의 지원을 받아 수행된 연구임 (NRF-2023S1A5B5A17083999)

**단독저자: 北京師範大學 歷史學院 博士研究生, classtoy@naver.com

in a Human-in-the-Loop workflow, in which the researcher provided requirements, error messages, and related inputs, while the AI agent iteratively revised and verified the code. The prompts, code, and sample data associated with this work are made available in the study's GitHub repository.
Keywords: *Shun Pao,* Shanghai Library, text data, Digital History, Human-in-the-Loop

1. 서론

《申報》(영문명: *Shun Pao*)는 찻잎과 직물을 취급하던 영국 상인 어니스트 메이저(Ernest Major)가 1872년 4월 30일 상하이에서 창간하여 1949년 5월 27일 중국인민해방군의 상하이 입성과 함께 정간되기까지 77년여간 간행된 중국어 일간지로, 그 발행 호수는 약 27,000여 호에 이른다. 중국에서 가장 이른 시기에 창간되었을 뿐 아니라, 중화인민공화국 수립 전까지의 근대 신문으로는 가장 오래 발행된 신문 중 하나다. 《신보》는 외국인이 발행하는 신문이었음에도 중국인 주필을 고용해 중국인의 관점을 반영하고, 중국 문인들로부터 독자 투고를 받거나, 다른 신문에 비해 가격을 저렴하게 책정하는 등의 경영 방침을 바탕으로 짧은 시간 내에 상업적 성공을 거두었다.[1] 이러한 초기 성공은 이후 《신보》가 상하이 지역 매체에 그치지 않고 주요 도시로 분관을 확장하면서 전국 단위 매체로 발돋움하는 동력이 되었고, 이는 동시기 창간된 신문 중 가장 오래 존속할 수 있는 기반이 되었다. 《신보》는 많은 돈을 들여 특파원을 고용해 역사적 사건의 경과를 상세히 전달했으며, 정치 사건 외에 경제, 문화, 과학기술 방면의 주요 소식을 전하는 데도 소홀하지 않았고, 나아가 전문 분야에 관한 부록도 다수 발간한 바 있어 중국근대사의 백과사전으로 평가받는다. 뿐만 아니라 그 지면 광고는 영화, 건축, 교통, 출판, 패션, 기호품, 연애관 등 사회문화사 분야에 풍부한 연구 소재를 제공하여, 상하이사 연구가 중국의 다른 어떤 도시사 연구보다도 많이 이루어질 수 있었던 주된 요인 가운데 하나로 꼽히기도 한다.[2] 이러한 점에서 《신보》는 중국근대사 연구에서 빼놓을 수 없는 중요한 사료라 할 수 있다.

이 논문은 중국 상하이도서관(上海圖書館) 《신보》 데이터베이스 텍스트 데이터 수집 스크립트의 작성 과정과 실제 수집 절차, 수집 데이터의 구조와 특징, 그리고 연구용 통합 데이터 구축을 위한 전처리 스크립트의 처리 절차와 결과물 구조를 설명한다. 상하이도서관 《신보》 데이터베이스는 베이징더홍과기유한공사(北京得泓科技有限公司)[3]가 제공하는 전문 검색 데이터베이스(全文檢索數據庫)로, 1872년 4월 30일 창간호부터 1949년 5월 27일 폐간호까지, 상하이판뿐 아니라 1937년 11월 일본군의 상하이 점령으로 인한 정간 도중 발간된 한커우판(漢口版)과 홍콩판(香港版) 기사까지 포함하는 총 2,893,427건, 약 42만 면의 데이터를 갖추고 있다.[4] 상하이도서관에 따르면, 이 데이터베이스에서는 텍스트 전문과 300dpi 이상의 원문 영인 이미지로 구성된 이중 플랫폼을 통해, 별도의 클라이언트 소프트웨어 설치 없이 열람, 검색, 텍스트 복사, 다운로드, 출력이 가능하다. 또한 섹션명, 표제, 저자, 본문, 광고 등의 텍스트에 대한 인공 검수를 거쳐 데이터 신뢰도를 높였다고 밝히고 있다.[5] 해당 데이터베이스는 상하이도서관 전문 서비스 포털(上海圖

書館專業服務門戶) [6] 메인 페이지 상단 메뉴바의 "资源导航"(영문 홈페이지 표기: "RESOURCE GUIDE") 탭 아래 위치한 "数据库导航"(영문 홈페이지 표기: "Database Guide") 또는 "数据库检索"(영문 홈페이지 표기: "Database") 메뉴를 클릭한 후, 검색창에 간체 중문으로 "申报"를 입력해 검색된 항목에서 "馆外访问"을 클릭하면, 도서관 계정 로그인을 거쳐 접근 가능하다.[7]

그림 1. 상하이도서관 《신보》 데이터베이스 진입 페이지

이하에서 다룰 데이터 수집 및 전처리 스크립트는 Windows 기반 로컬 환경에서 작성·실행하였다. Visual Studio Code 1.115.0을 편집 도구로, Python 3.11.9를 구현 언어로 사용했고, 실행 및 검증은 PowerShell을 통해 수행하였다. 코드 작성 과정에서는 OpenAI Codex 기반 코딩 에이전트(GPT-5.4)를 활용하였으며, 연구자가 요구 사항, 오류 메시지 등을 제시하고 이에 따라 수정과 검증을 반복하는 Human-in-the-Loop 방식으로 작업을 진행하였다. 작업 관련 프롬프트와 코드, 데이터 샘플은 이 연구 GitHub 저장소에 공개한다.[8]

2. 데이터 수집

2.1 수집 스크립트

상하이도서관 《신보》 데이터베이스는 접근 과정에서 대상 사이트가 통합 인증(OAuth) 기반의 로그인과 자동입력 방지 문자 입력을 요구하고, 기사 목록 페이지와 기사 본문을 포함하는 상세 페이지가 모두 동적으로 구성되어 있어, 정적 HTML 문서 수집 방식으로는 데이터 확보가 어렵다. 따라서 이 연구 코드 설계는 로그인 및 수집 목표 기사 목록

으로의 진입까지는 사용자가 진행하고, 그 다음 절차부터 자동화하는 반자동 방식을 채택하였다. 수집 스크립트의 초기 작성 단계에서는 기본적인 동작 설계가 이루어졌다. 코드를 시행하면 열리는 브라우저를 통해 사용자가 수동 로그인하여 수집 목표 기사 목록 페이지에 도달한 뒤 터미널에서 Enter 키를 입력하면, 자동화 코드가 브라우저를 제어해 목록 페이지와 상세 페이지를 순회하며 데이터를 수집하는 방식이다. 코딩 에이전트는 설계 과정에서 수집 대상 페이지의 구조를 검토한 후, 동적 페이지 전환, 조건부 대기, 예외적 페이지 구조에 대한 대응 등 기능을 더 안정적으로 구현할 수 있다는 점을 근거로, 기존에 널리 활용되어 온 브라우저 자동화 라이브러리 Selenium 대신 Playwright[9]의 사용을 제안했다. 또한 수동 로그인 이후 브라우저 상태를 유지한 채 데이터를 수집하는 반자동 작업 흐름에도 Playwright가 더 부합한다는 의견도 제시했다. 이 같은 에이전트의 답변을 참고하여, 수집 스크립트는 Playwright for Python 1.58.0 기반으로 Google Chrome 브라우저를 자동화하는 방식으로 구현되었다.

구현된 스크립트(파일명: collect_shenbao_text_chrome.py)의 기본적인 시행 순서는 다음과 같다.

① 사용자: 터미널에서 수집 스크립트를 실행한다.

② 수집 스크립트: 사용자에게 수집 대상 데이터를 식별할 레이블을 입력하도록 안내한다.

```
No --label/--resume-file/--resume-latest provided. Enter label to
continue:
```

③ 수집 스크립트: Chrome 브라우저를 실행하고, 상하이도서관 《신보》 데이터베이스 URL (https://z.library.sh.cn/http/80/77/30/1/10/yitlink/)에 접속한 뒤 로그인 페이지로 이동한다. 사용자에게 수동 조작 관련 안내 메시지를 출력한다.

```
Log in manually, move to the target results page, and set page size to 100.
When the current browser tab is ready for crawling, return here and press
Enter.
```

그림 2. 상하이도서관 통합 계정 로그인 페이지

그림 3. 《신보》 데이터베이스 초기 화면

④ 사용자: 상하이도서관 이용자 계정으로 수동 로그인한다.

⑤ 사용자: 키워드 검색이나 기간 설정 등을 통해 수집 대상 기사 목록의 첫 번째 페이지로 이동한다. 해당 데이터베이스의 중국어 텍스트는 번체 중문을 기반으로 하기 때문에, 검색어 역시 번체 중문으로 입력해야 한다. 간체 중문을 사용할 경우 검색 결과가 나타나지 않는다.

그림 4. 데이터베이스 검색 결과 목록 페이지

⑥ 사용자: 한 페이지에 100건의 기사가 표시되도록 목록 설정을 변경한다(초기 설정은 20건).

⑦ 사용자: 터미널로 돌아와 Enter 키를 입력한다.

```
Open pages in context: 1
[Page 1]
url=https://passport.library.sh.cn/oauth/authorize?response_type=code&scope=read&client_id=2046230678&redirect_uri=http%3a%2f%2fz.library.sh.cn%2fcallback%2fauthorization_code_callback&user_type=AC&state=123456
[Page 1] title_links=0
[Page 1] frames=1
  [Frame 1] url=https://z.library.sh.cn/go?url=http://10.1.30.77:80/
title_links=0
Detected title links in best frame: 100Starting crawl from frame URL:
https://z.library.sh.cn/http/80/10.1.30.77/yitlink/tm/index1.php
Page title:
Page 1: 100 result blocks
```

⑧ 수집 스크립트: 목록 페이지에서 첫 번째 기사의 목록 제목 정보를 읽어 수집 데이터에 기록한다.

⑨ 수집 스크립트: 첫 번째 기사 링크를 클릭하여 상세 페이지로 이동한다.

⑩ 수집 스크립트: 상세 페이지에서 텍스트를 추출하여 수집 데이터에 기록한다.

그림 5. 기사 상세 페이지

```
Collected page=1 item=1 title=1. {목록 페이지상의 기사 제목}
```

⑪ 수집 스크립트: 목록 페이지로 복귀한다.

⑫ 수집 스크립트: 목록 페이지에서 두 번째 기사에 대해 같은 작업을 수행한다(이후 ⑧부터 ⑪까지 반복).

⑬ 수집 스크립트: 현재 목록 페이지의 100번째 기사까지 수집을 마치면 다음 목록 페이지로 이동한다(이후 ⑧부터 ⑬까지 반복).

```
Page 2: 100 result blocks
Collected page=2 item=101 title=101. {목록 페이지상의 기사 제목}
```

⑭ 수집 스크립트: 수집 대상 기사 목록의 마지막 항목까지 처리를 마치고 다음 기사나 다음 페이지가 더 이상 없으면 작업을 종료한다.

```
Next page button not found on page {도달 페이지}. Stopping.
Current scope URL:
https://z.library.sh.cn/http/80/10.1.30.77/yitlink/tm/index1.php
```

⑮ 수집 스크립트: 수집 결과를 CSV 형식으로 저장하고, 작업 시간을 출력한다.

```
Saved {수집 기사 수} rows to
{로컬 작업경로}\shenbao\shenbao_textdata\shenbao_textdata_{label}_{첫 기사 일련번호}to{마지막 기사 일련번호}.csv
Elapsed time: 00m 00s
```

수집 대상 페이지의 구조와 관련해, 이 코드가 안정적으로 작동하기 위해서는 각 단계에서 다음 동작으로 넘어가는 조건을 명확히 설정해야 한다. 해당 데이터베이스는 기사 목록에서 기사 제목을 클릭할 때 별도의 새 창이 열리는 방식이 아니라, 동일 브라우저 탭 내부에서 프레임 구조를 통해 상세 페이지로 전환되는 방식으로 구현되어 있다. 다시 말해, 기사 본문은 목록 페이지와 분리된 독립 창이 아니라, 동일 탭 안의 상세 화면에 표시된다. 또한 기사 링크 역시 단순한 정적 URL과 달리 사용자의 실제 클릭 동작을 전제로 구성되어 있어, 목록 페이지를 거치지 않은 직접 접근으로는 본문 내용이 나타나지 않는다.

이 같은 페이지 구조에서는 사용자가 수동으로 로그인한 뒤 브라우저 상태를 유지한 채, 스크립트가 순차적으로 브라우저 상호작용을 수행하도록 구성할 필요가 있다. 동시에 기사 링크 클릭, 상세 화면 전환, 목록 화면 복귀 과정에는 로딩 지연이 수반되므로, 기사 링크 클릭 → 상세 페이지 데이터 추출 → 목록 페이지 복귀의 반복이 지나치게 빠르게 이루어지지 않도록 대기 조건을 설정해야 한다. 그러나 페이지 로드 시간은 인터넷 접속 상태나 데이터베이스 서버 응답 속도에 따라 크게 달라질 수 있으므로, 고정된 대기 시간을 설정하는 방식은 안정적이지 않다. 따라서 본 코드에서는 각 단계에서 수집 대상이 되는 HTML 요소가 실제로 나타나는지를 다음 동작의 조건으로 설정하고, 그 조건이 충족될 때에만 이후 절차를 진행하도록 하였다.[10]

한편 이 스크립트는 사용자가 수집 대상을 구분할 수 있도록 레이블(label) 입력을 기본 전제로 한다. 터미널에서 코드를 실행할 때 아래와 같이 레이블 관련 명령어를 별도로 지정하지 않으면, 코드 진행 순서 ②에서 제시한 바와 같이 수집 대상 식별을 위한 레이블 값을 입력하도록 안내한다. 입력된 레이블은 “shenbao_textdata_{label}_{첫 기사 일련번호}to{마지막 기사 일련번호}.csv” 형식의 출력 파일명에도 반영된다.

```
PS {로컬 작업경로}> python collect_shenbao_text_chrome.py --label {레이블 값}
```

이처럼 파일명에 포함된 레이블은 단지 사용자의 파일 식별만을 위한 것이 아니라, 수

집의 안정성을 확보하기 위한 장치이기도 하다. 수집이 장시간 이루어지는 경우, 페이지 전환 실패, 상세 페이지 로딩 실패 등 오류로 인한 중단을 완전히 방지하는 것은 불가능하다. 또한 수차례 실행을 통해 확인한 바, 로그인 세션은 최대 2시간까지만 유지되기 때문에, 1회 실행 시 수집할 수 있는 기사는 최대 2,000건 내외로 제한된다. 따라서 중간 저장과 작업 재개 기능을 구현할 필요가 있으며, 이때 레이블은 중간 저장된 파일을 재개 대상 파일로 식별하는 기준으로 사용될 수 있다.

중간 저장 기능은 수집 진행 도중 모종의 이유로 수집이 중단될 경우 직전까지 수집한 데이터를 CSV로 저장한 후 스크립트가 종료되도록 하는 것이다. 작업 재개 기능은 특정 수집 대상 목록에 대한 데이터 수집이 완료되지 않았을 때, 스크립트가 이전 작업에서 수집한 항목을 중복 수집하지 않고, 중단 지점까지 자동으로 이동한 뒤 수집을 재개하도록 하는 것이다.

재개 기능과 관련된 명령어로는 “--label {레이블 값}”, “--resume-latest”, “--resume-file {재개 대상 파일경로}”의 세 가지가 있다. 우선 “--label {레이블 값}” 명령어는 해당 레이블을 가진 기존 결과 파일이 없을 경우 새로운 수집을 시작하고, 결과물을 그 레이블이 붙은 파일로 저장한다. 반대로 동일한 레이블을 가진 기존 파일이 저장 경로에 존재할 경우, 스크립트는 이를 재개 대상으로 인식하고 그 파일에 기록된 마지막 기사의 일련번호부터 수집을 이어간다. “--resume-latest” 명령어는 결과 저장 경로에서 가장 최근 수정된 파일을 재개 대상으로 설정한다. 이때 “--label {레이블 값}”과 “--resume-latest”를 함께 사용하면, 해당 레이블에 속한 파일 중 가장 최근 파일을 재개 대상으로 탐색한다. “--resume-file {재개 대상 파일경로}” 명령어는 재개 대상 파일을 직접 지정하는 방식이다.

예를 들어 직전 작업에서 수집 대상 목록 중 240번 기사까지 처리된 상태에서 작업이 중단되었다면, 재개 명령어를 사용해 스크립트를 다시 실행할 경우 터미널에는 다음과 같은 메시지가 출력된다.

```
...
Page title:
Advancing from page 1 to page 2 for resume.
Advancing from page 2 to page 3 for resume.
Page 3: 100 result blocks
Collected page=3 item=240 title=240. {목록 페이지상의 기사 제목}
...
```

이 메시지는 수집 스크립트가 240번 기사가 위치한 지점에 도달하기 위해 수집 대상 목록 1페이지에서 2페이지로, 다시 2페이지에서 3페이지로 이동한 후, 3페이지에서 240번 기사를 찾아 수집을 이어 갔음을 의미한다.

그 밖에 안정성 확보를 위해 구현한 명령어로는 몇 건마다 중간 저장을 수행할 것인지

를 지정하는 “--save-every {숫자}”, 시험 실행이나 부분 수집을 위해 최대 수집 페이지 수를 제한하는 “--max-pages {숫자}”, 결과 저장 경로를 직접 지정하는 “--output-dir {경로}” 등이 있다.

수집 데이터의 저장 형식은 CSV(Comma-Separated Values, 쉼표로 구분된 값)로 설정했다. 수집 목표가 출처 페이지의 HTML 계층 구조 전체가 아니라, 제목 필드와 본문 필드의 텍스트 데이터에 한정되고, 이후 전처리 단계에서 텍스트로부터 추출·분리할 메타데이터 역시 중첩 구조를 반드시 필요로 하지는 않기 때문이다. 이런 경우 데이터를 행 단위로 정리해 기사 개체 1건당 1행으로 구성된 CSV 형식으로 구축해도 무방하다. 또한 수집 과정상 중간 저장-작업 재개 로직의 안정적 실행이나, 수집 이후의 오류 검수 및 분석용 파생 데이터 추출에도 CSV가 더 적합할 것으로 보인다.

Python 환경에서 CSV를 다룰 때 한 가지 유의할 점은 Python에 기본으로 들어있는 CSV 처리 모듈이 한 칸(필드)에 저장할 수 있는 최대 길이에 제한이 있다는 사실이다. “csv.reader”, “csv.DictReader”, “csv.writer”, “csv.DictWriter” 등 모듈에서 한 칸 최대 길이 기본값은 131,072자로, 별도의 설정 없이 작업을 진행할 때 한 칸 안의 문자열이 131,072자를 초과하면 작업 과정에서 다음과 같은 오류가 발생한다.

```
_csv.Error: field larger than field limit (131072)
```

지면 한계가 뚜렷한 신문 텍스트 특성상 문자 수가 기본값인 131,072자를 넘을 가능성은 낮지만, 실제 수집 과정 중 광고 기사 내의 텍스트가 이 값을 초과해 오류가 발생한 사례가 드물게 발견되었다. 해당 사례는 총 2건으로, 모두 광고 지면이었고, 각각 322,522자, 165,801자였다. 이는 광고 하나가 매우 긴 내용을 담고 있는 것이 아니라, 데이터베이스 구축 과정에서 한 광고 지면에 들어 있는 길고 짧은 광고들을 여러 개체로 분리하지 않고 하나의 기사 개체로 묶어 놓았기 때문에 생긴 현상이다.[11] 이러한 오류가 발생하지 않으려면 수집 스크립트 안에 아래와 같이 한 칸 최대 문자 수 제한을 올리는 모듈을 포함시켜야 한다. 최종 스크립트에서는 상한을 1,000,000자로 설정했고, 이후 수집 과정에서는 같은 문제가 발생하지 않았다.

```
csv.field_size_limit(1_000_000)
```

2.2 수집 결과 데이터

위와 같이 작성한 수집 스크립트를 사용하여 확보한 결과 데이터 사례는 다음과 같다.

- shenbao_textdata_lixian_1to7203.csv
- shenbao_textdata_xianfa_1to18648.csv
- shenbao_textdata_xianzheng_1to9906.csv
- shenbao_textdata_zhixian_1to4322.csv

공통 접두부인 “shenbao_textdata” 다음에 표시된 “lixian”, “xianfa”, “xianzheng”, “zhixian”은 모두 스크립트 실행 시 입력한 레이블 값이다. 각 데이터가 “立憲”, “憲法”, “憲政”, “制憲”을 검색어로 사용해 반환된 목록을 수집한 결과물임을 나타낸다. 파일명 말미의 “1to{숫자}”는 검색 결과 목록상 첫 번째 기사의 일련번호인 “1”과 마지막으로 수집 완료된 기사의 일련번호를 나타내며, 이를 통해 키워드별 수집량을 확인할 수 있다. 각 파일은 각 검색어에 대해 데이터베이스가 반환한 검색 결과를 모두 수집한 결과이므로, “{숫자}”는 해당 검색어를 포함하는 데이터베이스 내 기사의 규모를 나타낸다. 단, 수집 완료 후에는 CSV의 마지막 행 번호가 마지막 기사 일련번호에 헤더 행 1개를 더한 값과 일치하는지 확인하여, 수집 누락 유무를 확인해야 한다. 위 수집 사례 중에는 “xianfa” 데이터에서 1개의 누락이 확인되었고(일련번호: 9122), 해당 행은 원 데이터베이스에서 기사 데이터를 직접 확인한 뒤 열 구조에 맞춰 수동으로 입력해 보완하였다.

그림 6. 수집 CSV 데이터 예시(shenbao_textdata_lixian_1to7203.csv, Visual Studio Code 화면)

수집 데이터는 label, page, item_index, list_title, detail_url, publish_variant, date, issue_page, special_column, h1, lv1_div, content-box2, era_year, category, theme, collect_error의 16열로 구성된다. 각 열은 아래와 같은 특징을 갖는다.[12]

- label: 해당 행이 어떤 검색 조건 또는 수집 단위에서 확보된 기사인지를 식별하기 위해 수집 스크립트 실행 시 사용자가 입력한 레이블 값을 기록한 열이다.

- page: 해당 기사가 수집 당시 검색 결과 목록에서 위치하고 있던 페이지 번호를 기록한 열이다. 목록 페이지는 기사 출처 유형에 따라 《신보》 본지(상하이판) -> 광고면 -> 《신보》 한커우판 -> 《신보》 홍콩판 순으로 정렬되어 있고, 동일 출처 내에서는 시대 역순으로 정렬되어 늦은 시기의 기사일수록 작은 값을, 이른 시기의 기사일수록 큰 값을 갖는다.
- item_index: 검색 결과 전체에서 해당 기사가 갖는 일련번호를 기록한 열이다. 목록 페이지 제목 앞에 표시되는 "{숫자}." 형식의 문자열에서 추출된다. page 값과 마찬가지로 늦은 시기의 기사일수록 작은 값을, 이른 시기의 기사일수록 큰 값을 갖는다.
- list_title: 목록 페이지에 표시된 기사 제목을 저장한 열로, 목록상의 일련번호(item_index 값)를 포함하고 있다. 상세 페이지의 HTML 구조상 "#content-box_contentwrapper" 내부의 "h1" 요소에 대응한다.
- detail_url: 기사 제목을 클릭하여 상세 페이지에 진입했을 때 열린 기사 페이지의 실제 URL을 기록한 열이다. 기사별 고유 식별자(id), 기사 출처 유형 코드(qrynewstype), 검색 결과 일련번호(n) 등을 포함한다. 기사 출처 유형 코드는 《신보》 본지 기사를 가리키는 "SP", 광고면 기사를 가리키는 "SP_AD", 한커우판 기사를 가리키는 "SP_FH", 홍콩판 기사를 가리키는 "SP_HK" 네 가지 중 한 값을 갖는다.
- publish_variant: 상세 페이지 최상단 구획에 표시되는 기사 출처 유형을 나타내며, "申報", "申報_漢口版", "申報_香港版" 세 가지 중 한 값을 갖는다. 데이터베이스에서 광고면 기사로 분류된 행에는(detail_url 내 유형 코드 "qrynewstype=SP_AD") 해당 열의 값이 결측되어 있다.
- date: 상세 페이지 최상단 구획에서 "日期 : " 다음에 오는 "YYYY-MM-DD" 형식의 발행일자 정보를 저장한 열이다.
- issue_page: 상세 페이지 최상단 구획에서 "版次/卷期 : " 다음에 오는 "nn版" 형식의 지면 정보를 숫자 "nn"으로 저장한 열이다.
- special_column: 상세 페이지 최상단 구획에서 "專欄:" 다음에 오는 문자열을 저장한 열이다.
- h1: 상세 페이지 내 두 번째 구획, 즉 HTML 구조상 "#content-box_contentwrapper" 내부의 "h1" 요소에서 추출한 텍스트를 저장한 열이다. 일반적으로 해당 기사의 제목에 해당하며, 목록 페이지에 표시된 제목(list_title)과 달리 일련번호를 포함하지 않는다. 단, 제목이 없고 본문만 있는 기사의 경우 본문 내용이 해당 영역에 표시되기도 한다.
- lv1_div: 상세 페이지 내 세 번째 구획, 즉 HTML 구조상 "h1" 요소 바로 아래의 첫 번째 "div" 요소에서 추출한 텍스트를 저장한 열이다. 일반적으로 기사 제목과 본문 사이에 들어가는 보도 출처 정보를 포함한다. "本報訊"이 54.05%(21,663/40,079행)로 가장 많은 비중을 차지하며, "中央社", "新華社", "路透社"(로이터 통신) 등 통신사 명칭 등도 자주 등장한다. 다만, 이 영역은 상세 페이지에서 제목 영역인 "h1"과 본

문 영역인 “content-box2” 요소 사이에 위치하는데, 원 데이터베이스 구축 과정에서 괄호 표지를 기준으로 한 기계적 분절로 인해 일반적인 데이터 구조상 들어가지 않아야 할 문자열이 들어간 경우가 적지 않아, 반드시 보도 출처 정보를 담고 있다고 볼 수 없다.

- content-box2: 상세 페이지 내 네 번째 구획, 즉 HTML 구조상 “div class=content-box2” 요소에서 추출한 텍스트를 저장한 열이다. 이 열은 일반적으로 기사 본문을 포함하나, 제목이 없는 기사 중 본문 전체가 “h1” 요소에 탑재된 경우 문자열을 갖지 않아 결측값이 된다.
- era_year: 상세 페이지 내 다섯 번째 구획, 즉 HTML 구조상 “div class=content-box2” 요소 다음에 위치한 “div” 요소에서 “其他紀元：” 다음에 오는 “淸...年 日...年” 또는 “民國...年 日...年” 형식의 기년 정보를 저장한 열이다.
- category: 상세 페이지 내 다섯 번째 구획에서 “類別 : ” 다음에 오는 문자열을 저장하는 열이다.[13]
- theme: 상세 페이지 내 다섯 번째 구획에서 “主題 : ” 다음에 오는 문자열을 저장한 열이다. 해당 문자열은 원자료에서 유래하는 것이 아닌 데이터베이스가 자체 제공하는 주제별 기사 탐색 기능을 위해 부여된 주석이다.[14]
- collect_error: 상세 페이지 내 텍스트 데이터 수집에 실패한 경우 그 오류 메시지를 기록하는 열이다. 수집 과정 중 발생할 수 있는 데이터 누락 여부를 검토하기 위한 항목으로, “[ERROR] {오류 발생 사유}...”의 형식으로 출력된다.[15]

위 데이터 구조를 확정하기까지 두 차례의 스크립트 개선이 있었다. 최초 작성한 스크립트는 데이터를 page, item_index, title, publish, detail_url, text의 총 6열로 수집하는 구조를 갖고 있었다. 이 스크립트로 수집한 데이터에서는 크게 두 가지 문제점이 발견되었는데, 첫째는 복수의 검색 결과 목록을 수집한 후 이를 하나의 데이터셋으로 통합했을 때, 각 기사가 어떤 키워드 검색 결과로부터 수집되었는지를 확인할 수 없다는 것이었다. 이를 개선하기 위해 label 속성을 추가해 데이터 통합 후에도 각 기사를 어느 수집 조건(검색 키워드, 기간 등)에서 확보했는지 추적할 수 있도록 했다.

두 번째 문제는 title 열에서 발생했다. 최초 스크립트는 목록 페이지 내 제목 구획의 문자열을 title 열에 저장하도록 설계했으나, 수집된 title 값이 상세 페이지 제목 구획(“h1”)에 표시된 문자열의 앞쪽 100자 안팎(수집 데이터 기준 95~99자) 또는 검색어 주변 문맥 100자 안팎(수집 데이터 기준 97~105자)만 포함하는 사례가 확인되었다. 이는 상세 페이지 내 “h1” 문자열이 목록 페이지의 제목 표시 영역에서 수용할 수 있는 최대 글자 수(100자 내외)를 초과하는 경우, 목록 페이지에는 일부 문자열만 표시하도록 설정되어 있기 때문으로 보인다. 이 수집 방식으로는 데이터베이스가 제공하는 기사 제목을 온전히 수집할 수 없을 뿐 아니라, 서로 다른 검색 결과로부터 중복 수집된 동일 기사가 서로 다른 title 값을 갖는 경우도 발생할 수 있다. 이는 이후의 데이터 통합 단계에서 중복 수집된 같은 기사를 서로 다른 기사로 오인해 통합되지 않는 문제도 야기할 수 있다. 이

를 해결하기 위해 목록 페이지 제목 영역의 문자열은 list_title로 저장해 검수 및 보관 용도로만 사용하고, title 열에는 상세 페이지의 “h1” 요소에서 확보한 문자열을 저장하도록 수집 구조를 변경하였다.[16] 이 구조에서는 제목 데이터를 불러올 때마다 일련번호를 제거해야 하는 번거로움도 줄일 수 있다. 결과적으로 첫 번째 개선에서는 수집 결과가 label, page, item_index, list_title, publish, detail_url, title, text의 8열 구성으로 저장되도록 스크립트를 수정했다(파일명: collect_shenbao_textdata_chrome_ver2.py, 이하 Ver.2).[17]

한 차례 구조 개선을 거친 스크립트의 수집 결과에서도 두 가지 문제가 발견되었다. 하나는 일부 행의 text 열에 데이터베이스 서비스 업체 정보가 섞여 들어간 사례가 확인된 것이고, 다른 하나는 일부 행 title 열에 기사 제목만이 아니라 본문 전체 또는 일부가 포함된 경우가 적지 않다는 점이다. 앞서 1차 개선 과정에서 언급한 “h1” 영역의 문자열이 100자를 초과하는 경우는 대부분 실제로 기사 제목이 긴 것이 아니라, 이 같은 오탑재로 인해 발생한 것이었다.

본문 전체가 “h1”에 실린 기사는 대체로 제목이 없는 기사에서 자주 등장한다. 시기에 따른 지면 구성이나, 기사 특성에 따라 제목 없는 기사가 존재하는 것은 자연스러운 일이다. 다만 데이터베이스가 이런 기사를 처리하는 규칙이 일관되지 않은 점은 문제가 된다. 데이터베이스의 일반적인 구조에 따라 “h1”을 비우고 “content-box2”에 본문을 싣는 경우가 있는가 하면, 반대로 기사 본문 전체를 “h1”에 싣고 “content-box2”를 비운 사례도 상당 부분을 차지한다. 이 같은 탑재 방식 불일치는 본 연구가 구현하고자 하는 체계적 데이터 수집에 적지 않은 문제를 초래할 수 있다.

업체 정보가 text 열에 들어가는 현상은 이렇게 “content-box2”가 공백 상태가 되면서 발생한 대표적인 문제 사례다. 기존 스크립트는 기사 본문에 해당하는 “content-box2”가 비는 경우는 거의 없을 것으로 가정하고, 메타데이터 영역의 HTML 구조가 달라질 경우를 대비해 목표 DOM 요소에서 문자열을 수집하지 못하면 인근 요소를 수집하는 fallback 설정을 포함했다. 그런데 실제 수집을 진행한 결과, 메타정보는 대체로 일관된 구조로 표시되어 문제가 없었던 반면, 오히려 기사 본문이 “h1”에 배치되면서 “content-box2”가 비게 되고, 대신 footer 요소에 들어있는 업체 정보가 수집되는 현상이 적지 않게 발생했다. 이 문제를 해결하기 위해 fallback 설정을 제거하고, 수집 대상 요소가 문자열을 포함하지 않는 경우 해당 열을 결측으로 처리하도록 개선했다.

더 문제가 되는 것은 본문 일부가 “h1”으로 넘어가고, 나머지 일부가 “content-box2”에 남아 있는 경우다. 해당 기사군에서는 상세 페이지 “h1” 요소 아래, “content-box2” 요소 위에 위치한 “div” 요소에도 본래 들어가야 할 보도 출처(“本報訊”, “中央社”, “新華社”, “路透社” 등)가 아닌 다른 문자열이 나타나는 경우가 대부분이다. 원문 이미지와 대조해 보면, “div” 영역은 “h1” 마지막 내용과 “content-box2” 첫 내용 사이에 위치한 내용을 가리키는데, 이때 발견되는 사실은 원문에서 괄호 안에 위치한 내용이 “div” 영역에 탑재되었다는 점이다(그림 7). 이는 데이터베이스 구축 과정에서 괄호 표지를 전후한 문자열이 기계적으로 분절되어 서로 다른 DOM 구획에 배치되면서 나타난 현상으로 보인다.[18]

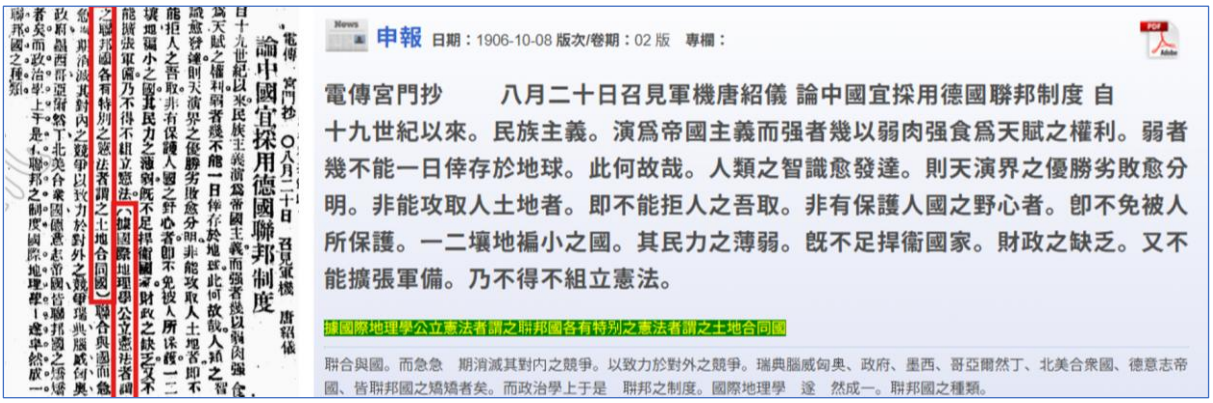

그림 7. 괄호 안 내용의 첫 번째 “div” 배치 사례
(좌: 원문 이미지, 우: 상세 페이지. 빨간색 강조는 필자.)

이처럼 각 HTML 요소가 정확히 제목, 보도 출처, 본문에 대응한다는 점이 보장되지 않는 상황에서, “h1” 문자열을 저장하는 열 제목을 title로, 첫 번째 “div” 문자열을 저장하는 열 제목을 publish 또는 source로, “content-box2” 문자열을 저장하는 열 제목을 text로 설정하는 것은 데이터 활용 시 해석상의 오인을 초래할 여지가 적지 않다. 따라서 수집 단계에서는 우선 각 문자열의 추출 위치를 반영하여 열 제목을 h1, lv1_div, content-box2로 변경하고, 그에 해당하는 영역의 문자열을 그대로 수집해 저장하도록 스크립트를 수정했다.

이에 더해 수집 후 전처리 부담을 줄이고자 초기 스크립트에서는 구획 내 문자열을 한 번에 수집하여 한 열에 서너 가지 메타데이터가 함께 저장되던 방식을 개선했다. 상세 페이지 내 DOM 영역 지정 외에 별도의 분할이 필요하지 않은 기사 출처(publish_variant), 발행일자(date), 지면(issue_page), 특집란(special_column), 기년(era_year), 유형(category), 주제(theme) 정보를 각각 따로 수집해 저장하도록 한 것이다. 수집 도중 발생할 수 있는 오류 메시지를 저장하는 열(collect_error)도 추가했다. 이를 통해 상술한 16열 구성으로 수집 데이터를 저장하는 스크립트가 완성되었다(파일명: collect_shenbao_textdata_chrome_ver3.py, 이하 Ver.3).

이 연구 GitHub 저장소에 공개한 수집 데이터 예시는 이상 두 차례 개선을 통해 수집 구조를 확정한 후 한 차례의 간단한 수집 방식 개선을 거친 Ver.3.1 수집 스크립트(파일명: collect_shenbao_textdata_chrome_ver3_1.py)[19]로 확보한 결과물이다. 수집 데이터는 데이터베이스상의 기사 개체를 기본 행 단위로 삼고, 텍스트 데이터를 속성별 열에 저장하는 구조를 갖추고 있지만, 검색어별 중복 수집, 즉시 사용 가능한 분석용 텍스트의 부재, 그리고 분석용 텍스트 생성 시 보정을 필요로 하는 일부 구조적 예외라는 문제를 여전히 포함하고 있다. 이어지는 전처리 단계에서는 수집 데이터를 통합하고, 중복 행을 제거하고, 분석에 사용할 수 있는 텍스트 단위를 만드는 작업이 필요하다.

3. 전처리 – 연구용 통합 데이터 제작

지금까지 살펴본 수집 데이터의 구조와 특징은 자동 수집 데이터를 곧바로 분석에 활용하기보다는, 일정한 정제와 통합 절차를 거칠 필요가 있음을 보여준다. 이 장에서는 원 수집 데이터를 연구용 통합 데이터로 재구성하기 위한 전처리 스크립트의 설계 배경과 실제 처리 절차, 단계별 산출물 구조를 기술한다. 전처리 작업은 크게 3단계로 구성된다. 첫째, 수집한 여러 원본 CSV 데이터를 하나의 데이터로 단순 병합한다. 둘째, 기사 개체를 식별하는 article_id를 기준으로 중복 수집된 행을 통합해 1행 1개체 구조를 만든다. 셋째, 분리 가능한 메타데이터를 별도 열로 추출하고, 분석에 사용할 수 있는 텍스트 열을 생성한다.

작업 진행 순서를 결정하는 과정에서 코딩 에이전트는 중복 제거를 먼저 진행할 경우 데이터 손실이 발생할 수 있다는 점을 근거로, 메타데이터 분리와 문자열 정제를 선행한 뒤 중복 행 통합을 마지막 3단계로 시행하는 방안을 제안한 바 있다. 그러나 본 데이터베이스는 detail_url에 포함된 id 값이 같으면 데이터베이스 내 동일한 기사 개체를 불러와 상세 페이지로 표시하는 구조이고, 실제 수집 결과물 검토에서도 중복 행 간의 수집 데이터가 대부분 일치하는 것을 확인할 수 있었다. 이는 중복 제거를 먼저 시행한 후에 문자열 정제를 진행해도 데이터 손실 문제가 거의 발생하지 않음을 의미한다. 따라서 본 연구 전처리 스크립트는 1단계 단순 병합, 2단계 중복 제거, 3단계 메타데이터 분리 및 분석용 텍스트 생성의 순서를 채택하였다.

이하에서는 전처리 단계의 전체 흐름을 개관하면서, 각 단계의 처리 내용과 생성 데이터의 구조를 살펴본다. 연구 과정에서 최초 작성한 전처리 스크립트는 수집 스크립트의 2차 개선(Ver.2→Ver.3) 이전에 설계한, Ver.2 수집 데이터에 대응하는 스크립트였다(파일명: shenbao_textdata_preprocess_combine.py). 그러나 Ver.2와 Ver.3 수집 데이터 간에 텍스트 추출 대상 DOM 구획과 열 구조의 차이가 적지 않아, 최신 수집 데이터에 맞춰 진행된 스크립트 개선은 기존 코드의 단순 개선이라기보다 사실상 새 스크립트를 설계한 것에 가깝다.[20] 따라서 여기에서는 전처리 스크립트의 버전별 수정 이력을 상세히 추적하기보다는, Ver.3 수집 데이터를 입력값으로 하는 두 번째 버전의 전처리 스크립트(파일명: shenbao_textdata_preprocess_combine_ver2.py)를 중심으로 논의를 진행한다.

전처리 스크립트를 실행하면 터미널에서 “Enter dataset_label for this combined dataset:”이 출력되고, 사용자가 원하는 데이터셋 레이블을 입력하면 스크립트가 시행된다. 기본 시행 순서는 다음과 같다.

① **1단계(stage1)**는 shenbao_textdata_{레이블 값}_1to{마지막 기사 일련번호} 형식의 파일명을 가진 수집 CSV의 단순 병합을 목적으로 한다. 이 단계에서는 원 수집 데이터의 값은 수정하지 않고, stage1_index 열만 추가하여 병합 이후의 추적을 위한 일련번호를 부여한다. 생성한 결과물은 shenbao_textdata_stage1_appended_rows_{dataset_label}.csv로 저장한다. 이 데이터는 수집 데이터의 16열에 stage1_index를 추가한 17열 구조를 가진다. 원 수집 데이터의 묶음인 stage1은 이후 단계에서 대표 행 선택과 원본 행 추적의 기준이

된다.

```
stage1_index,label,page,item_index,list_title,detail_url,publish_variant,
date,issue_page,special_column,h1,lv1_div,content_box2,era_year,category,
theme,collect_error
```

② **2단계(stage2)**는 각 기사 URL에 포함된 기사 식별자를 기준으로 한 중복 기사 통합을 목적으로 한다. 스크립트는 각 행에 기록된 detail_url에서 “id”와 “qrynewstype” 값을 추출해 각각 article_id, qrynewstype 열로 분리한다. “id” 값은 데이터베이스 내 기사 개체가 갖는 고유 식별자로, 여러 수집 단위에서 중복 수집된 기사 개체를 식별해 하나의 행으로 통합하는 기준으로 사용된다. “qrynewstype”은 “SP”, “SP_AD”, “SP_FH”, “SP_HK” 등의 값을 갖는데, 각각 “SP”는 《신보》 본지(상하이판) 일반 기사, “SP_AD”는 광고면 기사, “SP_FH”는 《신보》 한커우판 기사, “SP_HK”는 《신보》 홍콩판 기사를 지칭하며, 가장 기본적인 유형 분류에 활용할 수 있다. 다만 광고 기사 중에도 “SP_AD”로 표기되지 않은 경우가 다수 존재하기 때문에, 이 값만으로 광고 전체를 식별할 수 없다는 점은 유의할 필요가 있다.

③ 같은 article_id를 가진 행을 한 그룹으로 묶고, 각 그룹 안에서 대표행을 선택하여, 1개체 1행 구조의 2단계 데이터를 생성한다. collect_error가 비어 있는 행, issue_page가 비어 있지 않은 행, theme 길이가 더 긴 행, content_box2 길이가 더 긴 행, h1 길이가 더 긴 행, lv1_div 길이가 더 긴 행, 마지막으로 stage1_index가 더 작은 행의 우선순위에 따라 대표행이 될 수 있다. 이 같은 대표 행 선정 규칙은 단순히 제목이나 본문 텍스트 간의 비교에 그치지 않고, 수집 과정에서의 오류 발생 흔적이 없고, 데이터 충실도가 높은 행을 대표행으로 선정하기 위한 장치다. 대표 행 선택 사유는 select_reason 열에 1_no_error, 2_page_exist, 3_long_theme, 4_long_box2, 5_long_h1, 6_long_lv1_div, 7_small_index 중 하나로 기록된다.

④ 출처 정보 보존을 위한 열을 추가한다. dataset_label은 통합 데이터셋 전체에 부여한 공통 레이블이며, 본 연구를 통해 수집한 데이터에서는 “constitutional”을 사용하였다. source_labels에는 해당 기사 개체가 어떤 수집 라벨에서 확보되었는지를 “lixian ; xianfa”와 같은 형식으로 기록한다. stage1_indices에는 통합 대상이 된 1단계 데이터 중 원본 행들의 stage1_index 값을 “;”로 연결하여 기록한다. 또한 최종 대표행이 어느 원본 행에서 왔는지 확인할 수 있도록 representative_label, representative_item_index 열도 추가한다.

⑤ 같은 article_id를 가진 행들 사이에 값의 차이가 있는지 확인하여 기록한다. 검색 키워드가 다르더라도 값이 같아야 하는 publish_variant, date, issue_page, special_column, h1,

lv1_div, content_box2, era_year, category, theme, collect_error 등 11개 열을 비교하고, 이 중 하나라도 다른 열이 있으면 “collision=T”로, 차이가 있는 열 이름을 collision_columns 열에 표기한다. 값의 차이가 없는 경우 “collision=F”로 기록한다.

⑥ 생성 데이터를 저장한다. 저장 과정에서는 stage1의 17열 중 수집 시점까지만 의미를 갖는 목록 페이지 상의 정보, 즉 page(목록 페이지 번호)와 list_title(목록 페이지 제목) 두 열은 제거하고, stage1_index를 stage1_indices로, label을 representative_label로, item_index를 representative_item_index로 각각 대체한다. 여기에 article_id와 qrynewstype, dataset_label, source_labels, select_reason, collision, collision_columns의 일곱 열을 추가한 22열 구조 데이터를 shenbao_textdata_stage2_deduplicated_articles_{dataset_label}.csv로 저장한다.

```
dataset_label,source_labels,stage1_indices,representative_label,represent
ative_item_index,select_reason,article_id,qrynewstype,detail_url,publish_
variant,date,issue_page,special_column,h1,lv1_div,content_box2,era_year,c
ategory,theme,collect_error,collision,collision_columns
```

⑦ **3단계(stage3)**는 2단계에서 통합된 기사 모음을 연구용 데이터로 변환하는 것을 목적으로 한다. 우선 전체 행에 대한 정렬 규칙을 부여한다. 규칙의 우선순위는 date 오름차순, qrynewstype SP → SP_AD → SP_FH → SP_HK순, issue_page 오름차순, article_id 오름차순으로 한다. 이 중 date, qrynewstype, issue_page는 유의미한 순서 정보에 해당한다. article_id는 “A2012120747155”와 같은 형태를 띠는데, “A” 다음의 여덟 자리 숫자는 작업 일자, 뒤의 다섯 자리 숫자는 작업 시 부여된 일련 번호로 추정된다. 다만 다른 정보 값들과 일정한 비례 관계가 성립하지 않고, 원문 이미지 대조를 통해서도 지면상 기사 위치 등에 따른 번호 부여 규칙은 발견하지 못했다. 즉, article_id 오름차순은 다른 정렬 규칙과 달리 분석상 유의미한 정보를 포함하지 않는 임의 규칙으로, 다른 기준 값이 모두 동일할 때의 최종 결정 규칙(tiebreaker)으로만 기능한다. 정렬 이후에는 dataset_index 열을 추가하여 정렬 결과에 따라 번호를 부여한다. 달리 말하면, dataset_index가 작을수록 이른 시기, 클수록 늦은 시기에 발행된 기사이고, 같은 날 같은 신문에서 발행된 기사 간에는 번호가 작을수록 앞면, 클수록 뒷면에 실린 기사임을 나타낸다.

⑧ era_year 열에 포함된 중국 연호(“清...年” 또는 “民國...年” 문자열)와 일본 연호(“日...年” 문자열)를 분리하여 각각 chinese_era_year, japanese_era_year 열에 저장한다.

⑨ 분석용 텍스트를 저장하기 위한 analysis_text 열을 생성하고, h1, lv1_div, content_box2의 문자열을 순서대로 공백으로 이어 붙여 저장한다. 이 세 열은 텍스트 분석에서 직접 활용되는 핵심 데이터인 동시에, 다른 열에 비해 길고 비정형적인 문자열을 포함하다 보

니 그만큼 구조적 예외가 비교적 쉽게 발생한다. 따라서 analysis_text의 생성 과정은 단순히 세 열의 값을 기계적으로 결합하는 데 그치지 않고, 구조적 예외로 인한 데이터 중복과 오염을 줄여 이후 분석의 왜곡 가능성을 최소화하는 것을 목표로 한다. 최신 스크립트에서는 아래 다섯 가지 처리 규칙을 구현했고, 각 행의 analysis_text 생성 시 적용된 규칙은 analysis_text_rules 헤더를 가진 별도 열에 아래와 같이 저장한다.

1_drop_h1_for_classified_ad: special_column 값이 "分類廣告"인 경우, h1을 결합 대상에서 제외한다. 분류광고(分類廣告)는 일반 상업 광고와 달리 개인이나 소규모 단체가 싣는 생활정보 등에 관한 짤막한 광고로, 지면의 일정 구획에 항목별로 분류된 광고 문구가 모여 있는 형태다.[21] 광고 하나당 분량이 한두 줄 정도에 불과한 만큼, 데이터베이스에서도 각각의 광고를 개별 기사 개체로 탑재하는 대신, 한 지면 내의 광고를 모아 하나의 "分類廣告" 개체로 탑재한 것으로 보인다. 일반적으로 "分類廣告" 개체의 "h1"에는 데이터상 맨 위에 배치된 첫 광고의 제목이 들어가는데, 문제는 같은 문자열이 "content-box2"에도 들어가 있다는 점이다. 해당 문자열은 첫 번째 광고의 표제이므로 "content-box2"의 첫머리 문자열이 된다. 이 같은 중복이 실제 지면 배치를 반영하는 것인지 확인하고자 임의의 "分類廣告" 개체의 원문 이미지를 검토한 결과, 직접 확인한 경우에 한해 모든 개체에서 해당 문구는 중복 없이 한 번만 나타난다. 다시 말해, 이 같은 중복은 "分類廣告" 개체에 적용된 특수한 "h1" 탑재 방식으로 인한 것일 뿐, 실제 지면 배치와는 무관하다. 이러한 "分類廣告" 개체의 특성을 고려하지 않고 h1, lv1_div, content_box2를 공백으로 단순 연결하는 기본 결합 규칙을 적용하면, 생성된 텍스트 도입부에서 같은 문자열이 lv1_div를 사이에 두고 연달아 두 번 등장하게 된다. 이처럼 실재하지 않는 중복 출현을 보정하지 않을 경우, 출현 빈도 기반 분석에서 왜곡이 발생할 수 있다. 따라서 "分類廣告" 개체에 대해서는 h1을 결합에서 배제하는 것이 타당하다.

2_drop_benbaoxun: lv1_div 값이 "本報訊"인 경우, lv1_div를 결합 대상에서 제외한다. "本報訊"은 해당 기사가 다른 통신사 보도나 외부 기고, 지역 특파원 송고 기사가 아닌 본사 기자 작성 기사임을 나타내는 표지이다. 중복 제거된 수집 데이터 33,513건 중 lv1_div 값이 "本報訊"인 기사 개체는 18,511건으로 55.24%를 차지한다. 그런데 데이터상 "本報訊" 표지를 갖는 기사의 원문 이미지를 살펴보면, "本報訊"을 포함해 어떤 표지도 붙어있지 않은 사례가 적지 않게 존재하는 것을 확인할 수 있다. 이는 기사에 부여된 "本報訊" 문자열이 실제 원문에 "本報訊"이 표기된 기사뿐 아니라, 별도 표지가 없는 기사에도 일괄 부여된 것이 아닌가 짐작하게 한다.[22] 더구나 기사 본문 일부가 "h1"에 섞여 들어간 개체 중에도 lv1_div 값이 "本報訊"인 사례가 있어, 여기에 기본 결합 규칙을 적용하면 본문 중간에 실재하지 않는 "本報訊"이 삽입되는 문제가 발생한다. 이런 문제를 방지하기 위해 "本報訊" 개체에 대해서는 lv1_div를 결합에서 배제하는 규칙을 추가했다.

3_bracket_dedup: "special_column=分類廣告" 개체의 처리 규칙 설계 과정에서 h1과

content_box2 첫머리 문자열이 겹치는 사례를 검토하던 중, special_column이 비어 있는 개체에서도 문자열 이중 탑재가 발생한 사례를 발견하여, 이를 제거하는 규칙도 추가해야 했다. 이들 사례의 공통 특징은 첫째, h1의 전체 또는 마지막 문자열이 content_box2의 첫머리 문자열과 겹치고, 둘째, 중복 구간 양쪽 끝, 또는 한쪽 끝에 괄호 문자(“〔” 또는 “〕”)가 붙어 있다는 점이다. 이 같은 괄호 경계형 중복 현상은 앞서 언급한 괄호 표지 전후 문자열에 대한 기계적 분절이 비정상적으로 적용되어 발생한 것으로 추정된다. 다만 앞의 사례에서는 원문 이미지에 있던 괄호 문자가 문자열을 h1, lv1_div, content_box2로 나누는 분절 경계로 기능할 뿐 텍스트에까지 남지는 않았는데, 이 사례에서는 괄호 문자가 텍스트에 그대로 남아 있고, 괄호 안 문자열이 lv1_div에 탑재되는 대신 h1과 content_box2에 중복 탑재되고 있다. 또 하나의 특징은 텍스트에 남은 괄호 문자의 열고 닫는 방향이 원문 이미지 상의 방향과 반대로 되어 있다는 점이다(그림 8). 이 같은 중복 구간의 제거 규칙 설계에서는 h1과 content_box2 중 어느 쪽 문자열을 삭제할지를 결정해야 했는데, 둘 중 어느 쪽 문자열을 남기는 것이 원 데이터 특성에 더 부합하는지가 하나의 기준이 될 수 있다. 원문 이미지와 수집 데이터를 검토한 결과, 원문 표제가 content_box2에 오탑재되며 h1 문자열에 결손이 발생한 사례는 확인되지 않았고, 반대로 content_box2에 들어가야 했을 본문 내용이 h1으로 딸려 들어간 사례는 앞서도 이미 확인했듯 그 수량이 적지 않다. 이러한 데이터 특성에 비추어보면, content_box2의 문자열 결손을 감수하더라도 h1의 문자열을 보존하는 것이 더 타당하다. 따라서 괄호 경계형 중복 사례에서는 content_box2의 선두 중복 구간을 제거한 후 결합하는 규칙을 추가했다. 그림 8의 사례를 예로 들면, h1이 “滬道稟江督蘇撫文 〕呈送中國公學校舍地圖”이고 content_box2가 “呈送中國公學校舍地圖 〔 竊照中國公學學生張邦傑等…”일 때, 우선 content_box2에서 중복 구간인 “呈送中國公學校舍地圖”가 삭제된다. 이어서 h1, lv1_div, content_box2를 순서대로 결합해야 하지만, 여기에서는 lv1_div가 “本報訊”이므로, lv1_div를 제외한 나머지만 결합한다. 이렇게 생성된 analysis_text는 “滬道稟江督蘇撫文 〕呈送中國公學校舍地圖 〔 竊照中國公學學生張邦傑等…”로 시작하는 문자열을 갖고, analysis_text_rules에는 “2_drop_benbaoxun;3_bracket_dedup”과 같이 적용된 두 규칙이 함께 기록된다.

그림 8. 괄호 경계형 중복 탑재 사례(좌: 원문 이미지, 우: 상세 페이지. 빨간색 강조는 필자.)

4_delete_lv1_marker: 괄호 경계형 중복 탑재 사례 중 발견되는 예외 사례로, 중복 제거된 부분 직후의 문자열이 lv1_div와 동일한 경우가 존재한다. 괄호 경계형 중복 탑재가 발생한 개체는 lv1_div로 가야 할 구간이 h1과 content_box2에 남으면서 lv1_div는 사후에 일괄 부여된 것으로 보이는 "本報訊" 값을 갖는 것이 일반적인데, 예외 사례에서는 중복은 중복대로 남고, lv1_div에도 원래 들어가야 할 표지가 저장된다. 이때 content_box2 선두에 위치한 문자열은 "〔{lv1_div}〕"의 형태를 갖는데, 이는 이미 전방에서 괄호 경계형 중복 현상이 발생하여 괄호 문자를 경계로 h1, lv1_div, content_box2가 분할되는 규칙이 정상적으로 작동하지 않은 결과로 보인다(그림 9). 이런 사례의 처리는 우선 content_box2에서 h1과 겹치는 구간을 제거한 후, 남는 문자열 선두가 "〔{lv1_div}〕" 형태인 사례를 탐지해 제거하는 순서로 설계했다. 제거 대상을 판단할 때도 어느 쪽을 지우는 것이 원 데이터 구조에 더 부합하는지를 기준으로 하여, 일반적으로 content_box2 쪽의 "〔{lv1_div}〕" 문자열을 삭제하도록 규칙을 구현했다. 이 사례는 괄호 경계형 중복 제거 처리를 반드시 선행하므로, analysis_text_rules는 "3_bracket_dedup;4_delete_lv1_marker"로 기록된다.

그림 9. 중복 구간 뒤의 "〔{lv1_div}〕" 표지 사례
(좌: 원문 이미지, 우: 상세 페이지. 빨간색 강조는 필자.)

5_plain_merge: 위 네 가지 규칙 중 어느 것에도 해당하지 않는 경우, h1, lv1_div, content_box2를 순서대로 결합하는 기본 결합 규칙을 적용한다.

⑩ 이상의 과정으로 구조적 예외의 제한적 보정을 거쳐 결합한 분석용 텍스트를 포함하는 3단계 전처리 데이터를 shenbao_textdata_stage3_preprocessed_articles_{dataset_label}.csv로 저장한다. stage2의 22열 중 article_id와 qrynewstype의 출처가 된 detail_url 열은 제거하고, dataset_index, chinese_era_year, japanese_era_year, analysis_text, analysis_text_rules의 다섯 열을 추가한 26열 구조를 가진다.

```
dataset_label,dataset_index,source_labels,stage1_indices,representative_l
abel,representative_item_index,select_reason,article_id,qrynewstype,publi
```

```
sh_variant,date,issue_page,special_column,era_year,chinese_era_year,japan
ese_era_year,category,theme,collect_error,collision,collision_columns,h1,
lv1_div,content_box2,analysis_text,analysis_text_rules
```

4. 결론

지금까지 상하이도서관 《신보》 데이터베이스 텍스트 데이터 수집 스크립트의 작성 과정과 실제 수집 절차, 수집 데이터의 구조와 특징, 그리고 연구용 통합 데이터 구축을 위한 전처리 스크립트 작성 및 실제 처리 절차를 살펴보았다. 본문에서 기술한 초기 스크립트 설계와 이후 이어진 수차례의 코드 개선 과정에서는 AI 코딩 에이전트를 Human-in-the-Loop 방식으로 활용하였다. 이 과정을 통해 생성된 수집 및 전처리 산출물의 기본 정보는 다음과 같다.

표 1. 《신보》 데이터베이스 텍스트 데이터 수집 및 전처리 산출물의 기본 정보

구분	파일명	데이터 성격	행 수	열 수	시기 범위
원 수집 데이터	shenbao_textdata_zhixian_1to4322.csv	검색어: 制憲	4,322	16	1872-08-20 - 1949-05-26
	shenbao_textdata_xianzheng_1to9906.csv	검색어: 憲政	9,906	16	1875-08-31 - 1949-05-23
	shenbao_textdata_xianfa_1to18648.csv	검색어: 憲法	18,648	16	1873-09-02 - 1949-05-26
	shenbao_textdata_lixian_1to7203.csv	검색어: 立憲	7,203	16	1877-06-06 - 1949-05-08
전처리 1 단계	shenbao_textdata_stage1 _appended_rows_constitutional.csv	단순 병합	40,079	17	1872-08-20 - 1949-05-26
전처리 2 단계	shenbao_textdata_stage2 _deduplicated_articles_constitutional.csv	중복 제거	33,513	22	1872-08-20 - 1949-05-26
전처리 3 단계	shenbao_textdata_stage3 _preprocessed_articles_constitutional.csv	분석용 데이터	33,513	26	1872-08-20 - 1949-05-26

수집 스크립트는 자동화 수집에 앞서 키워드 검색, 시기 설정 등으로 수집 대상을 사전 한정할 수 있도록 설계되었다. 따라서 사용자는 자신의 연구 주제와 관련된 기사 목록을 먼저 구성한 뒤, 해당 목록을 대상으로 텍스트 데이터를 확보할 수 있다. 특히 《신보》와 같이 정치, 경제, 사회, 문화 등 다방면의 정보를 망라하는 매체로부터 자료를 수집할 경우, 이 같은 사전 필터링이 연구 목적에 맞는 자료군을 집중적으로 구성하는 데 도움이 될 수 있다.

당초 본 연구의 수집 목표는 《신보》 데이터베이스 내 기사의 전체 raw 데이터가 아니

라 그 안의 텍스트 데이터였으므로, 이 연구의 수집 데이터를 기반으로 수행할 수 있는 후속 작업 역시 다양한 텍스트 분석 방법론을 적용한 연구일 것이다. 이와 관련해 《신보》 텍스트 데이터가 번체 중문으로 이루어져 있음을 언급할 필요가 있다. 청말부터 중화민국 시기에 걸쳐 발행된 《신보》가 번체 중문으로 제공되는 것은 물론 자연스러운 일이다. 그러나 텍스트 분석의 질을 결정하는 핵심 요소 중 하나인 토큰화 모델은 간체 중문을 대상으로 한 것이 훨씬 다양하며, 모델의 미세 조정에서도 번체 중문 학습 데이터가 간체 중문 데이터만큼의 규모로 확보되기는 구조적으로 어렵다. 뿐만 아니라 《신보》가 발행되던 당시의 언어 습관은 오늘날의 중국어와 상당히 다르고, 같은《신보》안에서도 최대 77년에 이르는 시간차가 문체 차이로 뚜렷하게 드러난다. 중국근현대사 분야에서 디지털 역사학의 성패를 좌우하는 핵심 과제 가운데 하나는 결국 이러한 사료상의 언어와 현대 자연어처리 기술 사이의 간극을 어떻게 극복하느냐의 문제일 것이다.

---

[1] 《신보》의 창간 배경 및 과정, 초기 발전 등에 대해서는 上海書店《申報》影印組 編印(1983). <《申報》介紹>. 1-19; 方漢奇 主編(1992). <中國新聞事業通史>(第一卷). 中國人民大學出版社. 318-330; 劉家林(2012). <中國新聞史>. 武漢大學出版社. 75-90; 方漢奇(2012). <中國近代報刊史>(上冊). 山西教育出版社. 39-56; 심태식(2013). "19세기 말 중국 신문의 위상과 지식네트워크 - 초기 《申報》를 중심으로". <중국문학>. 75. 143-160. https://doi.org/10.21192/scll.75..201305.006; 花宏豔(2021). <《申報》的文人群體與文學譜系>. 商務印書館. 10-22 등에 자세하다. 여러 선행연구는 《신보》의 초기 성장 요인을 1861년 창간해 1872년 말 정간한 《上海新報》와의 대비를 통해 설명한다. 창간 초기 태평천국운동 소식을 전하며 성장한 《상해신보》는 《신보》 창간 이전까지 상하이의 유일한 중국어 신문이었으나, 상하이 조계 내 상인들을 주요 독자층으로 삼아 상업 관련 소식을 주로 다루었고, 더구나 영미권 출신 선교사들을 주필로 두어 기본적으로 서양인의 관점에서 보도가 이루어졌다. 그에 비해 《신보》는 현지 사회와의 상호작용을 바탕으로 폭넓은 독자층을 확보했고, 그 결과 상업적으로 더 성공할 수 있었다는 것이다. 광고 게재 측면에서도 《상해신보》에는 양행 광고가 주로 실린 것과 달리, 《신보》에는 극장, 식당, 오락시설 등 중국인들의 이목을 끌 수 있는 광고가 함께 실렸다는 점도 중요한 차이로 언급된다.

[2] 紹德(1983). "申報—近現代史料的寶庫". 上海書店《申報》影印組 編印. <《申報》介紹>. 135-136; 上海圖書館(2020). "舊時風月 : 從 《 申報 》 數據庫閱讀百年上海". https://mp.weixin.qq.com/s/LOBCEUkxwYffH4AygZsE-g. 일례로 하세봉은 정론지 위주였던 1920년대 이전 중국의 언론 지형 속에서도 상업지로서 성공을 거두었던 《신보》에 주목해, 그 도안 광고를 분석했다. 이 연구의 대상 시기는 지면 개혁을 통해 1면에 "論前廣告"를 싣기 시작한 1905년부터 5·4운동 무렵인 1919년까지로, 이 시기 동안 《신보》 지면은 절반 이상이 광고로 채워져 있었고, 그 중 도안광고는 4분의 1 정도로 추정된다. 하세봉은 당시 광고가 상정한 소비자층, 광고에서 나타나는 근대화-서구화 지향, 5·4운동으로 이어지는 이권 회수 운동과 함께 광고에 나타난 민족주의 등을 분석하는 한편, 도안 광고를 가능하게 한 동판 인쇄 도입이나 광고 대행업체가 부재하던 당시의 광고 도안가(디자이너)들에 대한 내용도 다루었다. 하세봉(2002). "近代中國의 新聞廣告 讀解 -辛亥革命 前後(1905-1919)의 『 申報 』 廣告-". <중국사연구>. 19. 235-270. https://www.riss.kr/link?id=A75365271.

[3] 베이징더훙과기유한공사는 학술 디지털 자원에 대한 조사 연구, 기획, 자문, 데이터 처리 및 제공 등 업무를 취급하는 회사로, 고서·근대 문헌·학술지 등의 디지털 데이터를 취급한다. 인문학 또는 역사학 분야에서 활용할 만한 데이터베이스로는 조룡고서데이터베이스(雕龍古籍數據庫), 《신보》를 비롯한 《大公報》, 《民國日報》, 《東方雜志》, 《晉察冀日報》, 《解放日報》 등 근대 매체의 전문 검색 데이터베이스(全文檢索數據庫), 중화인민공화국 시기 신문인 《人民日報》와 《光明日報》 원본 이미지 데이터베이스, 그리고 중국공산당사·중화인민공화국사 데이터베이스(中共黨史·中國國史數據庫) 등이 있다. http://www.libstage.com/.

[4] 《신보》는 중국근대사 연구의 핵심 자료 중 하나인 만큼, 근대 신문 잡지 자료를 취급하는 중국 내 주요 데이터베이스에서 그 원문 이미지를 제공하고 있으나, 방대한 분량으로 인해 텍스트 데이터를 구축해 제공하는 사례는 찾아보기 어렵다. 다만 한 GitHub 레포지토리(https://github.com/moss-on-stone/shenbao-txt)에서 《신보》 원문 텍스트를 제공하고 있는 것을 확인했으나, README.md 등의 설명에서 원자료의 구체적인 출처나 텍스트 확보 방식, 누락 범위 등을 제공하지 않아, 이를 연구 데이터로 사용할 경우 자료의 신뢰성과 활용 타당성 문제가 제기될 가능성이 적지 않다.

[5] 上海圖書館(2020). “舊時風月: 從《申報》數據庫閱讀百年上海”. https://mp.weixin.qq.com/s/LOBCEUkxwYffH4AygZsE-g.

[6] 상하이도서관 전문 서비스 포털(https://z.library.sh.cn)은 연구자를 위한 학술 자원 서비스로, 문헌 검색, 자원 탐색, 학문 분야별 서비스, 데이터베이스 원격 접속, 맞춤형 자원 추천 등의 기능을 제공한다. 상하이도서관 홈페이지(https://www.library.sh.cn) 상단 메뉴바의 “资源” 탭 아래 위치한 “数字资源” 메뉴를 클릭하는 방법으로도 전문 포털의 “数据库导航”(Database Guide) 페이지로 이동이 가능하다.

[7] 상하이도서관 계정(열람증)은 온라인으로도 발급이 가능하나, 외국인의 경우 가입 과정에서 중국 휴대전화 번호를 필요로 하고, 도서관에 직접 방문하여 이용증 발급 창구에서 열람증 활성화 절차를 거쳐야만 디지털 자원의 관외 이용이 가능하다. 자세한 내용은 “上海圖書館（上海科學技術情報研究所）讀者辦證須知”. https://www.library.sh.cn/guide/banzheng을 참고.

[8] https://github.com/GeonjoonBae/shlib-shenbao-dataset-workflow. 저장소는 수집 스크립트, 전처리 스크립트, 예외 데이터 검토 스크립트, AI 코딩 에이전트와의 작업 대화 기록, 키워드별 수집 데이터 샘플, 단계별 전처리 결과 샘플로 구성된다. 상세한 디렉터리 구조는 저장소의 README.md를 참조.

[9] Playwright는 2020년 마이크로소프트가 공개한 웹 브라우저 자동화 라이브러리로, JavaScript, Python, Java, .NET 등 여러 언어 환경에서 사용할 수 있으며, Chromium, Firefox, WebKit 계열 브라우저를 대상으로 자동화 기능을 제공한다. 해당 라이브러리 정보는 Playwright for Python 공식 웹페이지 (https://playwright.dev/python/docs/intro)와 GitHub 레포지토리 (https://github.com/microsoft/playwright-python) 등을 참고.

[10] 스크립트에 적용한 대기 설정은 다음과 같다. 페이지 이동 시 다음 동작에서 문제가 생기지 않도록 “wait_for_load_state( )”를 활용해 이동한 페이지의 기본 HTML 파싱 완료(“domcontentloaded”, timeout=3000)와 브라우저 네트워크 통신 안정(“networkidle”, timeout=5000)을 기다린다. 두 조건이 충족되면 이후 페이지 내에 수집 목표 HTML 요소가 나타나는지를 0.2초 간격으로 반복 확인하고, 해당 요소가 나타나면 수집한다. 한 목록 페이지에 표시된 기사 100건의 수집이 완료되고 다음 100건 목록으로 넘어갈 때는, 페이지 안정화를 위한 1초의 대기 시간을 두었다. 여기에 더해 페이지 전환이 원활히 이루어지지 않았을 경우의 대비책으로 60초의 최대 대기 시간을 설정했다. (MAX_RESULT_WAIT_MS = 60000)

[11] 이 중에는 두 면에 걸친 기사를 한 개체로 처리한 사례도 적지 않게 존재한다. 이는 데이터베이스 구축 과정에서 기사 개체를 구분하는 기준이 일관되지 않았을 가능성을 보여준다. 따라서 이 데이터베이스상의 한 개체가 항상 온전한 하나의 개별 기사를 의미한다는 전제는 성립하기 어렵다. 이러한 특성 때문에 본 수집 데이터를 기사 한 건 한 건이 분석 단위로 사용되는 TF-IDF 나 Doc2Vec 등 분석에 곧장 활용하는 것은 적절하지 않으며, 특정 키워드를 중심으로 한 문맥 단위나 개별 기사 단위로 분리하는 절차가 필요하다. 반면 지면상의 인접 맥락이나 여러 단신이 병치된 구조 자체를 보존하는 것이 유리한 분석 방법도 있다. 예컨대 코퍼스 구축, 글자·어휘 단위의 동시 출현 맥락 분석, 특정 용어가 등장하는 지면상의 주변 담론을 살피는 작업에서는 이러한 구조가 오히려 의미 있는 맥락 정보를 제공할 수 있다.

[12] 이하 수집 및 전처리 스크립트 작성 및 시행 과정에 대한 설명에서 원 데이터베이스의 DOM 요소를 지칭할 때는 큰 따옴표를 사용해 표기하고(“h1”, “div”, “content-box2” 등), 수집 데이터의 열 제목을 지칭할 때는 따옴표 없이 표기한다(h1, lv1-div, content-box2 등).

[13] 본 연구의 수집 데이터 총 40,079건 중 “類別：” 표지가 등장한 기사는 총 2,565건이지만, 실제로 후행 문자열이 존재하는 사례는 한 건도 없었다. 그러나 전체 데이터베이스 내에 해당 문자열을 갖는 개체가 단 한 건도 없다고 확신할 수는 없는 만큼, 해당 항목을 수집하는 기능을 제거하지 않고 유지했다.

[14] 데이터베이스 로그인 후 검색 페이지 좌측 패널에는 “年度”와 “主題”의 두 개 탭이 배치되어 있는데, 이 중에서 “主題” 탭을 클릭하면 주제별 탐색을 위한 계층형 목록이 나타난다. 목록에는 [+] 아이콘이 붙어 있는 상위 주제 항목이 제시되는데, 상위 주제의 글자 부분을 클릭하면 해당 주제에 관한 한 문단 정도의 설명이 표시되고, [+] 아이콘을 클릭하면 하위 주제 목록이 펼쳐진다. 전체 주제 분류는 총 47개 상위 주제와 122개의 하위 주제로 구성되어 있다.

[15] 본 연구의 수집 데이터 총 40,079행은 모두 collect_error 값을 갖지 않는다. 이는 《신보》 데이터베이스의 기본 탑재 구조가 비교적 일정하고, 수집 스크립트가 해당 구조 내에서 발생할 수 있는 예외 유형을 상당 부분 수용할 수 있는 방식으로 작성되었음을 의미한다.

[16] 수집 데이터에서 동일 행에 기록된 list_title과 title의 문자열을 비교해보면, 완전히 일치하는 경우가 64.81%, 공백이나 구두점 위치만 다른 경우까지 포함하면 총 31,011건으로, 전체 40,079건의 77.38%에 해당한다. 일치하지 않는 사례는 총 9,068건으로 22.63%를 차지하는데, 대부분 본문에서 설명한 상세 페이지 내 “h1” 문자열이 100자 내외를 초과하여 목록 페이지에서 생략형으로 표시된 경우에 해당한다. 한편 일치 사례 중 134건은 양쪽 모두 비어 있는 경우로, 1905년부터 1915년 사이, 특히 1906년(12건), 1909년(29건), 1910년(24건)에 집중되어 있으며, 황제 조서(上諭), 각 지역에서 전달된 단신, 광고 등을 포함한다. 그중에는 두 면에 걸쳐 인쇄된 기사의 두 번째 면 부분이 앞부분 내용과 별도 개체로 탑재되면서 제목란이 비워진 사례도 포함되어 있다.

[17] 1차 스크립트 개선 작업은 최초 작성한 스크립트를 별도 파일로 보존하지 않은 채 스크립트를 직접 수정하는 방식으로 이루어져, 현재는 ver1 코드가 남아 있지 않고 ver2 코드부터 확인 가능함을 밝혀둔다.

[18] 이 같은 분절은 《신보》 지면이 일반적으로 기사 제목에 이어 괄호 안에 보도 출처를 표기한 뒤 본문을 배치하는 편집 방식을 취한다는 가정 아래 일부 작업자에 의해 일괄 적용된 것으로 추측된다.

[19] 해당 개선은 Ver.3 수집 데이터 내 일부 행 lv1_div 에서 확인된 문자열 누락을 해결하기 위해 이루어졌다. 《신보》 데이터베이스는 사용자가 입력한 검색어를 쉽게 식별할 수 있도록 해당 글자에 강조 표시(span style="color:yellow;background-color:green")를 씌우는데, 기존 스크립트는 “h1” 아래 첫 번째 “div” 안의 직계 텍스트만 읽어 오도록 설계되어 있어, 효과가 적용된 부분이 누락되었던 것이다. 다른 DOM 요소에서는 이 같은 span 누락 문제가 나타나지 않았는데, 이는 lv1_div 에 저장할 문자열을 담고 있는 “div” 요소의 HTML 구조상 위치가 다른 요소들과 다르기 때문이다. 해당 “div”는 “content-box2” 등의 상위 컨테이너이기 때문에, 별도 설정 없이 이 요소를 통째로 읽을 경우 그 하위 요소까지 수집하게 된다. 따라서 이 요소에 대해서만 예외적으로 하위 “div”까지 탐색하는 대신 그 바깥에 위치한 텍스트만 불러오도록 스크립트를 설계한 것이다. 스크립트 개선은 강조 표시를 위해 삽입된 “span” 요소 내부의 문자열까지 함께 수집하되, 그 아래 이어지는 하위 “div” 요소는 수집하지 않도록 추출 방식을 수정하는 방향으로 이루어졌다.

[20] Ver.2 스크립트 작성 당시에는 “h1” 구획이 제목, “content-box2” 구획이 본문을 가리킨다는 전제를 아직 폐기하지 않은 상황에서, 중점은 한 열에 저장된 여러 메타데이터 문자열의 분해(parsing)에 있었다. 반면에 Ver.3 스크립트는 수집 단계에서부터 메타데이터를 가능한 한 분리 수집하여 개별 열에 저장하는 구조로 개선된 만큼 메타데이터 분해 기능은 축소되었고, DOM 영역별로 수집해 h1, lv1_div, content-box2 로 분할 저장한 기사 제목부터 본문까지의 텍스트를 하나의 분석용 데이터로 재구성하는 기능에 중점을 둔다.

[21] 분류광고에 자주 실리는 내용으로는 구인·구직, 임대·매매, 분실 신고, 모집 공고, 공개 구애·구혼 등이 있었다. 《신보》는 중국에서 이 같은 형식의 광고란을 최초로 도입한 매체로, 분류광고의 성행은 판로 확대와 구독 수요 증가로 이어져 《신보》의 상업적 성장에 기여했다고

평가받는다. 《신보》 분류광고에 대한 내용은 林升棟(1998). “《申報》分類廣告研究”. <新聞大學>. 98(秋). 73-75. https://doi.org/10.20050/j.cnki.xwdx.1998.03.018 을 참고.

[22] 수집 데이터에서 “本報訊” 기사의 비중을 살펴보면, 1872 년 창간부터 1904 년까지의 기사 개체 1,360 건 중 lv1_div 가 “本報訊”인 개체는 1,351 건(99.34%)으로, 심지어 모든 기사가 “本報訊”인 연도도 존재하지만, 해당 기사들의 원문 이미지를 확인해보면 대부분 어떤 표지도 붙어있지 않은 것을 알 수 있다. 한편 1905 년을 기점으로 “本報訊” 아닌 기사 비중의 단절적 증가가 나타나는데, 이때부터 폐간 시점인 1949 년까지 연간 비중이 20%대에서 70%대를 오간다. 이런 변화는 1905 년 2 월 7 일부터 공식적으로 진행된 《신보》의 지면 개편(改版)과 무관하지 않아 보인다. 당시 개편 요지를 담은 “신문 업무 정돈 12 조(整頓報務十二條)”가 여러 차례 지면에 게재되었는데, 일반 전보(郵電)가 아닌 본사 전용 전보(專電)를 통한 신속 보도, 외국 신문의 번역 게재, 취재원 및 특파원 초빙 확대, 독자 투고 확대 등 내용을 포함한다. 이에 따라 제목 아래나 본문 말미에 보도 출처나 기고자 정보를 기재하는 식의 유의미한 보도 표지 사용이 확대되면서, 표지 없는 기사에 일괄 부여되던 “本報訊” 비중도 자연스럽게 감소한 것으로 추정된다. 《신보》의 1905 년 지면 개편에 관해서는 武占江(2013). “論《申報》1905 年改版——兼論中國新聞史兩個系統的互動關係”. <西南民族大學學報(人文社會科學版)>. 2013(1). 180-185. https://doi.org/10.3969/j.issn.1004-3926.2013.01.029 를 참고.

참고문헌

1. 연구 데이터 저장소

배건준. “GitHub shlib-shenbao-dataset-workflow”. https://github.com/GeonjoonBae/shlib-shenbao-dataset-workflow.

2. 단행본

劉家林(2012). <中國新聞史>. 武漢大學出版社.

方漢奇 主編(1992). <中國新聞事業通史> 第一卷. 中國人民大學出版社.

方漢奇(2012). <中國近代報刊史>上冊. 山西教育出版社.

上海書店《申報》影印組 編印(1983). <《申報》介紹>. 上海書店.

花宏豔(2021). <《申報》的文人群體與文學譜系>. 商務印書館.

3. 논문

林升棟(1998). “《申報》分類廣告研究”. <新聞大學>. 98(秋). https://doi.org/10.20050/j.cnki.xwdx.1998.03.018.

武占江(2013). “論《申报》1905年改版——兼論中國新聞史兩個系統的互動關係”. <西南民族大學學報(人文社會科學版)>. 2013(1). https://doi.org/10.3969/j.issn.1004-3926.2013.01.029.

심태식(2013). “19세기 말 중국 신문의 위상과 지식 네트워크 – 초기 《申報》를 중심으로”. <중국문학>. 75. https://doi.org/10.21192/scll.75..201305.006.

하세봉(2002). “近代中國의 新聞廣告 讀解 ―辛亥革命 前後(1905-1919)의 『申報』廣告―”. <중국사연구>. 19. https://www.riss.kr/link?id=A75365271.

4. 웹페이지

北京得泓科技有限公司. http://www.libstage.com.

上海圖書館. https://www.library.sh.cn.

上海圖書館專題服務門戶. https://z.library.sh.cn.

上海圖書館(2020). “舊時風月：從《申報》數據庫閱讀百年上海”. https://mp.weixin.qq.com/s/LOBCEUkxwYffH4AygZsE-g.

Microsoft. “GitHub playwright-python”. https://github.com/microsoft/playwright-python.

Moss on Stone. “GitHub shenbao-txt”. https://github.com/moss-on-stone/shenbao-txt.

“Playwright for Python”. https://playwright.dev/python/docs/intro.

부록. 연구 산출물 및 데이터셋 구성

shlib-shenbao-dataset-workflow/
├ collect_shenbao_textdata_chrome_ver2.py
├ collect_shenbao_textdata_chrome_ver3.py
├ collect_shenbao_textdata_chrome_ver3_1.py
├ shenbao_textdata_preprocess_combine.py
├ shenbao_textdata_preprocess_combine_ver2.py
├ ai_coding_agent_dialogues/
  │ ├ 01_coding_1_상하이도서관 신보 데이터 크롤링 코드 작성.md
  │ ├ 01_coding_2_기사 페이지의 html 구조가 상이한 경우를 대비한 코드 수정.md
  │ ├ 02_preprocess_1_예외 데이터 식별.md
  │ ├ 02_preprocess_2_통합 csv 제작.md
  │ ├ 03_pipeline-revision_1_기존 코드의 fallback 기능으로 인한 오수집 개선.md
  │ ├ 03_pipeline-revision_2_lv1_div의 수집 로직 개선.md
  │ ├ 03_pipeline-revision_3_수집 구조 개선에 따른 전처리-통합 코드 수정.md
  │ ├ 10_data-profile_1_데이터 구조 및 필드별 충실도 검토.md
  │ ├ 10_data-profile_2_중복 개체 간 데이터 일치.md
  │ ├ ...
  │ ├ 10_data-profile_10_分類廣告 개체의 구조적 특징.md
  │ └ 10_data-profile_11_전처리 결과물의 특성과 column별 분포 특징.md
└ shenbao_textdata/
  │ ├ shenbao_textdata_lixian_1to7203.csv
  │ ├ shenbao_textdata_xianfa_1to18648.csv
  │ ├ shenbao_textdata_xianzheng_1to9906.csv
  │ ├ shenbao_textdata_zhixian_1to4322.csv
  └ preprocess/
    ├ shenbao_textdata_stage1_appended_rows_constitutional.csv
    ├ shenbao_textdata_stage2_deduplicated_articles_constitutional.csv
    └ shenbao_textdata_stage3_preprocessed_articles_constitutional.csv