파이썬을 이용한 프레임내 웹 페이지 스크래핑 기법

윤수진* · 승리 · 우영운

동의대학교

A Scraping Method of In-Frame Web Sources Using Python

Sujin Yun* · Li Seung · Young Woon Woo

Dong-eui University

E-mail : yun990701@naver.com / ywwoo@deu.ac.kr

요 약

이 논문에서는 일반적인 웹 접근 방법으로 접근하기 어려운 프레임 내 웹 페이지의 데이터를 프로그램에 의해 자동으로 수집하기 위한 세부 주소 확보 기법을 제안하였다. 제안한 세부 주소 확보 기법과 HTML 실렉터를 활용할 수 있는 Python 언어와 Beautiful Soup 라이브러리를 이용하여 여러 페이지로 작성되어 있는 게시판 텍스트 데이터를 자동으로 모두 수집할 수 있었다. 제안한 기법을 활용하여 어떠한 형태의 주소 형식으로 되어 있는 웹 페이지들에 대해서도 Python 웹스크래핑 프로그램에 의해 자동으로 대량의 데이터를 수집할 수 있으며, 이를 통해 빅데이터 분석에 활용될 수 있을 것으로 예상한다.

ABSTRACT

In this paper, we proposed a detailed address acquisition scheme for automatically collecting data of a web page in a frame that is difficult to access by a general web access method. Using the Python language and the Beautiful Soup library, which can utilize the proposed address resolution technique and the HTML selector, we were able to automatically collect all the bulletin board text data written in several pages. By using the proposed method, we can collect large amount of data automatically by Python web scraping program for web pages of any form of address, and we expect that it can be used for big data analysis.

키워드

Web scraping, Python, Beautiful soup, HTML selector, Big data

Ⅰ. 서 론

최근 웹 문서로부터 대량의 데이터를 자동으로 수집하여 빅 데이터 분석에 활용하는 사례가 늘고 있다[1][2]. 이 경우에 웹 페이지의 내용을 직접적으로 보여줄 수 있는 주소가 확보될 경우에는 자동화된 프로그램으로 그 주소를 직접 방문하여 필요한 데이터를 수집할 수 있다. 그러나 수집하려는 데이터가 존재하는 웹 페이지가 프레임 등으로 포함되어 있어 해당 웹 페이지에 대한 세부 웹 주소가 웹 브라우저 주소창에 직접 나타나지 않는 경우 등에는 그 주소를 이용한 웹 스크래핑에 어려움을 겪을 수 있다[3].

사용자가 직접 마우스를 이용하여 한 페이지씩 접근하여 웹 페이지 내의 텍스트 데이터를 수동으로 수집하고자 하는 경우에는 문제가 되지 않지만, Python 프로그램 등을 활용하여 자동으로 모든 페이지를 방문하면서 대량의 텍스트 데이터를 추출하기 위해서는 중요한 문제이다.

이 논문에서는 프레임에 포함되어 있는 세부 웹 페이지의 데이터를 자동으로 수집하기 위한 세부 웹 페이지의 주소를 확인하는 방법과, 이 주소를

* speaker

이용하여 웹 페이지 데이터를 자동으로 수집하기 위한 HTML 실렉터(selector) 기능과 Python 웹 스크래핑 프로그램[4]의 활용 방안 기법을 제시한다.

II. 제안 기법

이 논문에서는 “아이사랑”이라 불리는 아이사랑 보육포탈 홈페이지에 게시중인 게시판 데이터를 수집하기 위한 기법을 제시한다[3]. 이 사이트는 그림 1과 같이 프레임 구조를 이용하여 세부 페이지들을 표시하는 구조를 가지고 있다.

```
▶<frame name="mainFrame" id="mainFrame"
title="아이사랑보육포털" src="/cpin/
main1.jsp" scrolling="auto" frameborder=
"0" marginwidth="0" marginheight="0"
noresize="noresize">…</frame>
▶<noframes title="noframes">…
</noframes>
</frameset>
```

그림 1. 프레임 구조를 갖는 페이지 소스

이 논문에서는 이상의 구조를 갖는 “아이사랑” 웹 페이지에서 ‘상담실 -> 육아 상담 -> 자주하는 질문’ 게시판의 데이터를 수집하고자 하였는데, 이 부분까지 메뉴를 클릭하는 방식으로 들어가면 그림 2와 같이 주소창의 주소가 전혀 변경되지 않고 내용이 나타남을 알 수 있다.

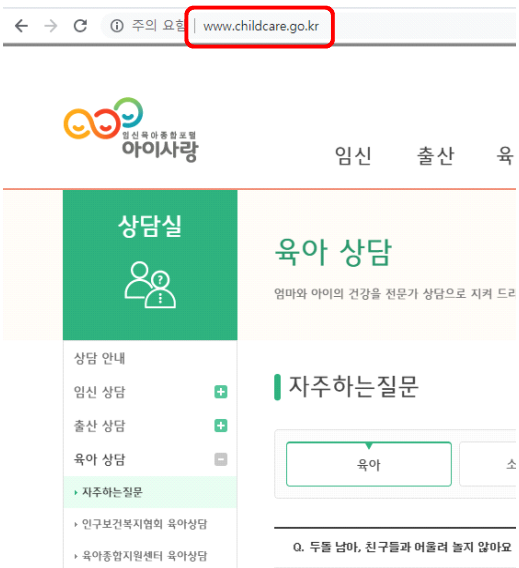

그림 2. 동일 주소에서 프레임 내 세부 웹 페이지 접근 예

따라서 이런 식으로 접근하게 되면 수집하려는 데이터가 존재하는 세부 웹 페이지의 실제 주소를 파악할 수 없기 때문에 웹 소스를 분석하여 그림 3과 같이 프레임으로 불러 들이는 웹 페이지의 세부 경로와 파일명을 확인하여 그림 4와 같이 그 세부 경로와 파일명으로 접속을 변경하도록 한다.

```
▶<frame name="mainFrame" id=
"mainFrame" title="아이사랑보육포
털" src="/cpin/main1.jsp"
scrolling="auto" frameborder="0"
marginwidth="0" marginheight="0"
noresize="noresize">…</frame>
```

그림 3. 프레임 내 웹페이지 세부 경로와 파일명

그림 4. 변경된 웹 페이지 접근 주소

그림 4와 같이 주소가 변경된 상태에서 다시 ‘상담실 -> 육아 상담 -> 자주하는 질문’ 게시판으로 접근하게 되면 그림 5와 같은 형식으로 주소창이 바뀌는 것을 확인할 수 있다.

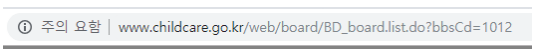

그림 5. 변경된 게시판 접근 주소

그림 5와 같이 변경됨으로써 수집하려는 정보가 게시되어 있는 게시판에 접근할 수 있게 되었다. 그러나 이 상태로는 한 페이지에 15개의 게시물만 나타나 있기 때문에 그림 6과 같이 게시물이 15개를 넘어가는 경우에는 그 이후의 페이지에서 나타나는 데이터들을 수집할 수 없다.

그림 6. 여러 페이지로 분리된 게시판 구조

이와 같은 경우에는 게시판의 2번째 페이지를 수동으로 먼저 접근하게 되면 그림 7과 같이 주소가 변경되는 것을 확인할 수 있다.

http://www.childcare.go.kr/web/board/BD_board.list.do?seq=&bbsCd=1012&pageType=&showSummaryYn=N&delDesc=&q_currPage=2&q_sortName=&q_sortOrder=

그림 7. 게시판의 각 페이지를 명시하는 구체화된 웹 페이지 주소

이상과 같은 과정을 거쳐 그림 7과 같이 최종적으로 획득된 세부 웹 페이지 주소에서 페이지 값을 1부터 마지막 페이지 값까지 자동으로 변경하면서 각 페이지의 게시판에 게시되어 있는 데이터들을 수집할 수 있게 된다.

III. 수행 결과

2장에서 제안한 프레임 내 웹 페이지 세부 주소 확인 기법을 활용하여 Python 웹 스크래핑 프로그램을 작성하였다. 작성된 프로그램의 코드 일부는 그림 8과 같다. 이 논문에서 작성한 웹 스크래핑 프로그램은 BeautifulSoup 라이브러리[5]를 이용하여 작성하였으며 각 게시판의 질문과 대답 영역을 추출하기 위해 사용한 실렉터 정보는 그림 9와 같다.

```
for page in range(1,5):
    visit='http://www.childcare.go.kr/web/'+\
    'board/BD_board.list.do?seq=&bbsCd=1012'+\
    '&pageType=&showSummaryYn=N&delDesc+\
    '=&q_currPage='+str(page)+'&q_sortName'+\
    '=&q_sortOrder='

    r = requests.get(visit)
    html = r.text
    soup = bs(html, 'html.parser')
```

그림 8. 파이썬 웹 스크래핑 코드 일부

```
질문:
'#dataForm > div > ul > li > ul > li > ul > li
> pre'

대답:
'#dataForm > div > ul > li > ul > li > pre'
```

그림 9. Q&A 추출을 위한 HTML 실렉터 정보

IV. 결 론

이 논문에서는 일반적인 웹 문서 접근 방법으로는 세부 페이지에 자동으로 접근하기 어려운 프레임 내 웹 페이지의 데이터를 수집하기 위하여, 세부 페이지 주소를 획득하는 방법을 제안하고 Python 웹 스크래핑 프로그램 기능과 HTML 실렉터를 활용하는 기법을 제안하였다. 제안한 기법을 활용하여 “아이사랑” 웹 사이트에서 여러 페이지로 분리되어 있는 게시판 데이터를 모두 수집할 수 있음을 확인하였다.

References

[1] K. W. Cho, S, K Bae and Y. W. Woo, “Analysis on Topic Trends and Topic Modeling of KSHSM Journal Papers using Text Mining,” *The Korean Journal of Health Service Management*, vol. 11, no. 4, pp. 213-224, Dec. 2017.

[2] Y. S. Kim and C. W. Seo, “A Study on the Analysis of Text Data Using Web Scraping of Cloud Computing,” *Proceedings of The Institute of Electronics and Information Engineering Summer Conference*, pp. 1736-1775, Jun. 2018.

[3] I-sarang, Comprehensive Pregnancy and Infant Care Website. [Internet]. Available: http://www.childcare.go.kr/.
[4] R. Mitchell, *Web Scraping with Python: Collecting Data from the Modern Web*, 1st edition, Sebastopol, CA:O'Reilly Media, Inc., 2015.
[5] Beautiful Soup Documentation [Internet]. Available: https://www.crummy.com/software/BeautifulSoup/bs4/doc/.