JKIICE Journal of the Korea Institute of Information and Communication Engineering

한국정보통신학회논문지 Vol. 27, No. 1: 1~7, Jan. 2023

심층 웹 문서 자동 수집을 위한 크롤링 알고리즘 설계 및 실험

강윤정[1] · 이민혜[1] · 원동현[1*]

Crawling algorithm design and experiment for automatic deep web document collection

Yun-Jeong Kang[1] · Min-Hye Lee[1] · Dong-Hyun Won[1*]

[1*]Assistant Professor, Division of Liberal Arts, Wonkwang University, Iksan, 54538 Korea

요 약

심층 웹 수집은 검색 양식에 질의어를 입력하고 응답 결과를 수집하는 것을 의미한다. 심층 웹이 가진 정보는 정적으로 구성되는 표면 웹보다 약 450～550배 이상의 정보를 가지고 있을 것으로 추산한다. 정적인 방식에서는 웹페이지가 새로 고쳐지기 전까지 변화된 정보를 보여주지 못한다. 동적 웹페이지 방식은 실시간으로 필요한 정보가 갱신되어 웹페이지를 새로 불러오지 않아도 실시간 정보 제공이 가능한 장점이 있지만, 일반적인 크롤러는 갱신된 정보에 접근하는 데 어려움이 있다. 따라서 이들 심층 웹에 있는 정보들을 크롤러를 이용해 자동으로 수집할 방안이 필요하다. 이에 본 논문은 스크립트를 일반적인 링크로 활용하는 방법을 제안하였으며, 이를 위해 클라이언트 스크립트를 일반 URL처럼 활용이 가능한 알고리즘을 제안하고 실험하였다. 제안된 알고리즘은, 검색 양식에 데이터를 입력하는 일반적인 방법 대신 메뉴 탐색 및 스크립트 실행으로 웹 정보를 수집하는 데 중점을 두었다.

ABSTRACT

Deep web collection means entering a query in a search form and collecting response results. It is estimated that the information possessed by the deep web has about 450 to 550 times more information than the statically constructed surface web. The static method does not show the changed information until the web page is refreshed, but the dynamic web page method updates the necessary information in real time and provides real-time information without reloading the web page, but crawler has difficulty accessing the updated information. Therefore, there is a need for a way to automatically collect information on these deep webs using a crawler. Therefore, this paper proposes a method of utilizing scripts as general links, and for this purpose, an algorithm that can utilize client scripts like regular URLs is proposed and experimented. The proposed algorithm focused on collecting web information by menu navigation and script execution instead of the usual method of entering data into search forms.

키워드 : 심층 웹, 웹 크롤링 , 자동 수집, 웹 정보, 웹 링크

Keywords : Deep Web, Web crawling, Automatic Collection, Web information, Web Link

Received 24 November 2022, **Revised** 1 December 2022, **Accepted** 15 December 2022

*** Corresponding Author** Dong-hyun Won (E-mail:dhwon79@wku.ac.kr, Tel:+82-63-850-6297)
Assistant Professor, Division of Liberal Arts, Wonkwang University, Iksan, 54538 Korea

Open Access http://doi.org/10.6109/jkiice.2023.27.1.1 print ISSN: 2234-4772 online ISSN: 2288-4165

Ⅰ. 서 론

심층 웹 수집은 검색 양식에 입력된 질의어의 결과를 동적으로 생성하여 결과를 수집하는 것을 말한다[1]. 월드 와이드 웹에서 심층 웹 정보 수집은 까다로우면서도 중요한 문제이다. 정적인 방식에서는 웹페이지가 새로 고쳐지기 전까지 변화된 정보를 반영할 수 없다. 동적 웹은 웹페이지를 새로 불러오지 않아도 실시간 정보 제공이 가능하다. 이러한 변화로 웹은 검색 양식으로 정보를 제공함과 동시에 메뉴에서 제공되는 정보들을 동적으로 변경하거나 추가하여 사용자에게 맞춤 정보를 제공할 수 있는 지능형 웹으로 변화하고 있다[2]. 이러한 변화로, 웹 정보 수집 도구인 크롤러는 다음과 같은 문제에 직면하게 된다.

첫째, 동적 웹은 수시로 변하기 때문에 웹 크롤러의 방문주기에 따라 웹의 변화된 정보를 수집하지 못할 가능성이 있다. 즉, 크롤러가 웹페이지를 방문했을 때 특정 시간 이후 정보가 갱신되기 때문에 기존 크롤러는 페이지의 전체 내용을 수집하지 못한다.

둘째, 동적으로 제작된 게시판은 일반적으로 웹페이지의 주소가 변경되지 않은 상태로 게시물을 로드한다. 예를 들어 게시물을 클릭하면 관련 스크립트가 실행되고 서버로부터 전송받은 데이터를 웹페이지에 반영하고 웹페이지 내용을 변경한다.

셋째, 일반적인 웹 크롤러는 URL을 수집하고 방문한다. 스크립트 형태의 주소는 URL 형태가 아니므로 크롤러는 스크립트를 URL로 수집하지 못한다.

상기한 어려움으로 수집하기 어려운 형태의 웹페이지를 심층 웹이라고 한다. 심층 웹이 가진 정보는 정적으로 구성되는 표면 웹보다 약 450～550배 이상의 정보를 가지고 있을 것으로 추산하고 있으며[2], 따라서 이들 심층 웹에 있는 정보들을 크롤러를 이용해 자동으로 수집할 방안이 필요하다.

이에 본 논문은 심층 웹 기록물을 자동으로 수집할 수 있는 크롤링 알고리즘을 제안하고자 한다.

Ⅱ. 관련 연구

2.1. 웹 크롤러

웹 크롤러는 웹 스파이더 또는 웹 로봇이라고도 불리며, 문서를 분석, 인덱스화, 검색 및 마이닝할 수 있도록 데이터 소스에서 문서를 수집하는 웹 문서 수집 프로그램이다[3].

Table. 1 Web Crawler Types and Collection Methods

Type	Collection Method
Focused Crawler	• Focus on collecting topics selected as important • Collect pages or topics selected as particularly important for collecting each page
Incremental Crawler	• Collect information that has not been collected since the last crawl by crawling the web document specified in the content source being crawled • To add only changes to existing crawl information
Distributed Crawler	• Multiple servers running crawlers on distributed system-based crawlers simultaneously collect web documents as a way to solve time-consuming problems in the process of collecting web documents • Collect web documents by placing a central server and managing each crawler server
Parallel Crawler	• Developed as a way to address the difficulty of collecting entire web pages with only one process or thread as the web grows in size • Collect large volumes of web pages quickly using multiple processes and threads

크롤러는 웹 정보 수집을 위해 웹페이지에 방문해야 하고 이를 위해 방문하기 위한 URL이 있어야 한다. 크롤러에 모든 웹 주소를 입력할 수 없으므로, 웹 크롤링 프로그램은 Seed URL에서 시작하여 방문한 웹페이지를 분석하여 방문하지 않은 모든 URL을 추출한다. 새로 방문한 URL 중에서 방문하지 않은 URL에 대한 정보를 추가로 추출하는 과정을 반복하여 Seed URL과 관련한 모든 웹페이지를 방문한다[1,2].

표 1과 같이 크롤러는 사이트 방문이나 크롤러 실행 방식에 따라 집중 크롤러(Focused Crawler), 증분 크롤러(Incremental Crawler), 분산 크롤러(Distributed Crawler), 병렬 크롤러(Parallel Crawler)로 구분지어 진다[3].

2.2. 심층 웹 크롤러

심층 웹에서 제공하는 콘텐츠는 검색엔진으로 쉽게 찾을 수 없다. 일반적인 검색엔진의 웹 크롤러는 심층 웹의 숨겨진 콘텐츠를 탐색하지는 못하기 때문이다[4]. 반면에 많은 정보가 심층 웹에서 정보를 제공하고 있고 이러한 웹 환경에서 정보를 수집하기 위한 다양한 연구

가 진행되고 있다. Deep bot[5]은 미니 웹브라우저라는 클라이언트 스크립트 실행 도구를 내장하여 서버와의 세션 유지 및 클라이언트의 스크립트 실행이 가능하게 하여 자료를 수집한다. HiWE[6]는 웹 질의어 인터페이스에 숨겨진 데이터를 추출하기 위해 입력 폼을 분석하고 입력 폼에 값을 전달하는 과정으로 심층 웹을 수집하는 크롤러이다. Incremental Web Crawler[7] 는 심층웹이 제공하는 정보의 변화에 즉시 반영된 결과를 저장하기 위한 크롤러이다. Incremental Web Crawler는 웹페이지에 방문하는 방문주기를 확률적으로 정하고 웹페이지 변화 주기를 계산하여 재방문에 대한 최적의 값을 적용하는 방식이다. 표 2은 심층 웹 수집을 위한 크롤러와 수집 방식이다.

Table. 2 Types and Collection Methods of Deep Web Crawlers

Type	Collection Method
Deepbot	• Collect data by embedding a client script execution tool called a mini-web browser to maintain sessions with the server and enable the client to run scripts
HiWE	• Collect deep webs by analyzing input forms and passing values to input forms to extract data hidden in the web query interface
Incremental Web Crawler	• Crawler to store results immediately reflected in changes in information provided by the deep web • A method of probabilistic determination of the visit cycle at which crawlers visit a webpage and calculating the change cycle of the webpage to apply the optimal value for revisiting

Ⅲ. 심층 웹 문서 수집 방안 제안

3.1. 기존 심층 웹 크롤러 방식의 어려움

심층 웹에서 문서를 수집하기 위해 기존 크롤러들은 질의어를 관리한다. 웹사이트에서 수집하고자 하는 정보에 맞게 질의어를 전송하고 전송 결과를 수집한다. 질의어 기반 크롤러는 질의어를 어떻게 관리하느냐에 따라 크롤링 결과가 달라질 수 있다. 질의어에 따라서 출력되는 정보가 다르기도 하고 중복된 정보가 출력되기도 하기 때문이다.

표 3에서 정리한 것처럼 각 크롤러는 입력 폼에 입력할 키워드들을 관리해야 하며, 상황에 따라서는 많은 키

Table. 3 Weakness of Deep Web Crawler

Type	Weakness
Deepbot	• Difficulty managing data to fill in a form • Difficult to solve when there are many input variables in the webpage input form • Many query language must be used to collect deep web data • Must run crawler frequently
HiWE	• People need to intervene to put data in input forms. • Collecting deep webs requires managing many query languages.
Incremental Web Crawler	• Keyword Management Required • Need a lot of storage space to manage your data • Difficulty indexing data after collection

워드를 입력하여 자료를 수집해야 한다. 각 경우 특정 키워드 입력으로 얻게 되는 자료가 달라서 사람이 직접 키워드를 분석하여 최적의 키워드를 관리하고 유지하는 것이 필요하다. 또한, 모든 단어를 입력 폼에 입력하는 방법은 현실적으로 불가능하며 많은 중복 결과를 가져올 수 있어서 키워드 관리 및 수집 정책에 대한 여러 어려움이 발생하여 사람의 개입 없이 자동화한 웹 수집 알고리즘을 적용하는 데 어려움이 있다[8].

3.2. 제안 알고리즘

본 논문이 제안하는 심층 웹 기록물 자동 수집 과정은 그림 1과 같다. 방문해야 할 웹페이지의 Seed URL을 방문하여 문서를 분석한다. 분석한 문서에서 링크 정보를 분류하여 다음 방문을 위해 Seed URL에 추가하고 Seed

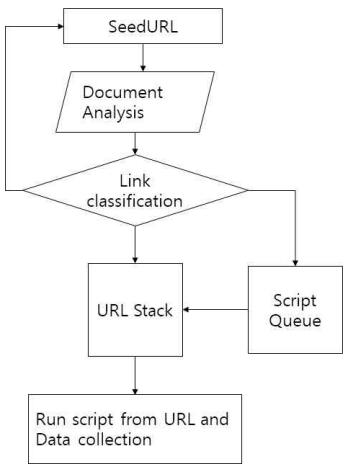


Fig. 1 Crawling process

URL 수집이 끝나면 순차적으로 URL에 접속하여 다시 링크 관련 정적 주소와 스크립트를 수집한다. 수집한 주소는 방문 여부를 확인하여 URL Stack에 저장하고, 스크립트 정보는 Script Queue에 저장한다.

주소 체계는 웹브라우저 주소창에서 실행되지 않고, 웹브라우저가 내장하고 있는 스크립트 인터프린터가 실행함으로, 자동 수집알고리즘은 웹 문서의 페이지 이동 관련 스크립트를 추출하는 것으로 시작된다. 웹 문서 스크립트가 실행되면 스크립트가 동적으로 웹 기록물을 생성하는 동안 대기하였다가 생성된 심층 웹 기록물을 수집한다.

이동할 페이지 주소는 A 태그의 Href 속성에 표기되어 있거나 A 태그 내의 Onclick 속성 뒤에 표기되어 있다. 일반적으로 스크립트로 실행될 때 A 태그의 href 속성은 “http://”로 시작하지 않으며 자바스크립트의 함수 형태를 보인다. 스크립트를 수집할 페이지에서 관련 명령어들을 수집하여 하나씩 실행하면서 결과를 저장하는 과정을 수행한다. 수집된 스크립트는 수집된 문서에서만 실행할 수 있으므로 스크립트를 실행하기 위해서는 어떤 페이지에서 수집되었으며 실행되어야 하는지 정보를 관리할 필요가 있다. 그림 2는 스크립트를 수집한 웹페이지에서 해당 스크립트를 실행하고 링크를 추출하는 과정을 보여준다. 먼저 웹페이지에서 추출된 ‘Web Page URL’은 ‘URL Stack’에 저장된다. 또한 해당 웹페이지에서 수집된 스크립트는 Script Queue에 저장하고 URL Stack에 저장된 URL과 연결한다. 가장 최근 방문한 웹사이트의 주소 먼저 방문하여 문서를 수집하고 Script Queue에 저장된 스크립트들을 실행하면서 변경된 정보들을 수집한다.

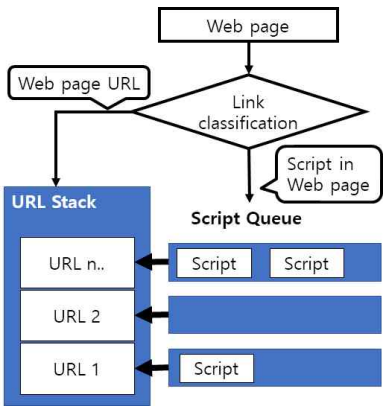


Fig. 2 Run a script and extract links

저장된 스크립트는 입력 순서대로 실행된다. 스크립트는 내장된 브라우저에서 ‘URL Stack’에서 추출한 페이지가 로드되면 실행하게 되고, 스크립트 실행으로 페이지 내용이 변경되면 변경된 문서를 분석한다. 다시 방문할 필요가 있는 페이지는 ‘URL Stack’에 저장되며 아직 실행하지 않은 스크립트가 발견되면 ‘스크립트 Queue’로 저장된다. ‘스크립트 Queue’가 모두 실행되면 ‘URL Stack’에서 다음 URL을 가져오게 되며 이러한 과정을 ‘수집 Page URL Stack’에 정보가 없을 때까지 반복한다.

심층 웹 문서는 실행 결과에 따라 주소가 변경되기도 하고 변경되지 않기도 한다. 정적인 주소의 경우 주소가 변경되나, 스크립트가 실행될 때는 웹 주소가 변경되지 않기도 한다. 이러한 특성으로 웹페이지가 변경되었는지 확인하기 위해서는 주소의 변경 여부뿐만 아니라 콘텐츠의 변경 내용도 확인해야 한다.

이와 같은 문제를 해결하기 위해 웹페이지의 변화와 주소 변경 여부에 따라 모든 스크립트가 실행되었는지 판단하였다. 표 3과 같이 웹페이지 주소나 내용 변경 여부에 따라 스크립트를 실행할 것인지 페이지를 이동할 것인지를 판단하게 된다. 먼저, 주소와 내용이 모두 변경된 경우는 새로운 페이지에서 새로운 내용이 로드된 경우로 해당 내용을 새로운 페이지와 같이 취급하여 주소와 스크립트를 추출한다.

다음으로 주소의 내용이 변경되었는데 내용이 변경되지 않거나 스크립트 실행 후에도 주소나 내용의 변경이 없는 경우에는 게시판과 같은 형태로 구성된 페이지의 마지막 페이지에 도달하였음에도 해당 스크립트를 실행하는 경우이다.

Table. 2 Document analysis and processing conditions after script execution

Address change	Content Change	Process
○	○	• Execute the following script in script Queue • Save New Address to Collect Page URL Stack • Depending on the depth of the visit setting, you may not visit
○	×	• Determined that the last page has been reached • Load the following address from the collection page URL Stack

Address change	Content Change	Process
×	○	• Execute the following script in script Queue
×	×	• Determined that the last page has been reached • Load the following address from the collection page URL Stack

이 같은 경우는 해당 스크립트를 실행한 것으로 하고 스크립트 Queue에서 다음 스크립트를 실행한다. 마지막으로 주소가 변경되지 않았는데 내용이 변경된 경우, 페이지에서 스크립트가 실행된 경우임으로 내용을 수집하고 다음 스크립트를 실행한다.

Ⅳ. 실험 및 평가

파이썬 라이브러리인 Beautiful Soup[9]과 Chrome Driver[10]를 설정하고 파이썬 코드로 알고리즘을 구현하였다. 그리고 수집된 정보는 MongoDB[11]에 저장하였다.

크롤러 실험 대상은 행정안전부 https://www.mois.go.kr 홈페이지를 대상으로 하였다. 행정안전부 홈페이지는 공공성을 가지고 있으며, 정적인 페이지와 동적인 페이지로 구성되어 본 논문에서 제안하는 알고리즘을 실험하기 적합하였다.

행정안전부를 Seed URL로 입력하고 수집된 정보에서 주소 정보를 분석하여 정적 주소 형태의 Seed URL과 스크립트 형태의 페이지 이동 스크립트 등을 수집하였다. 표3은 자동으로 수집된 URL과 해당 URL에서 동작하는 스크립트의 예시이다.

그림 3과 같이 행정안전부의 정보를 수집하여 문서

Table. 3 URL and Script queue Example (www.mois.go.kr)

URL	https://www.mois.go.kr/frt/a02/localGovernmentList.do

Script Queue	javascript:fnLink('https://www.data.go.kr/tcs/eds/ctm/selectContestDataList.do','Y');
	javascript:fnLink('https://www.mois.go.kr/frt/bbs/type001/commonSelectBoardArticle.do?bbsId=BBSMSTR_000000000015&nttId=87063','Y');
	javascript:fnLink('https://www.mois.go.kr/frt/bbs/type001/commonSelectBoardArticle.do?bbsId=BBSMSTR_000000000015&nttId=86790','Y');
	javascript:fnLink('https://www.mois.go.kr/frt/bbs/type001/commonSelectBoardArticle.do?bbsId=BBSMSTR_000000000015&nttId=86790','Y');

제목과 주소, 스크립트를 수집하여 MongoDB에 저장하고 수집된 URL에 접속하여 정보를 수집하였다.

Fig. 3 Data stored in MongoDB

전체 수집된 데이터는 표 4와 같다. 먼저 스크립트가 없는 정적인 형태의 URL은 5,693개였으며 스크립트가 포함된 URL 및 스크립트는 22,346개였다. 모든 URL과 스크립트를 실행하여 약 365MB의 정보를 수집하였으며 주요 단어를 WordCloud[12]로 분석한 결과 그림 4와 같이 업무, 안전, 정보, 관리 공개 등의 단어들이 높은 빈도로 나타났다.

표4를 통해 확인할 수 있듯이 정적인 주소를 가진 페이지보다 스크립트 기반의 웹페이지가 약 4배 많음을 확인할 수 있었다. 이러한 정보들은 검색엔진에서 접근하기 어렵고, 스크립트 실행 특성상 주소가 같으면서 내용이 달라서 원본 정보에 접근하기 어렵다. 이러한 문제들로 인해 기존 키워드 기반 심층 웹 수집 프로그램은 자동화에 어려움이 있었다.

본 논문에서 제안하는 알고리즘은 키워드와 관련 없이 사이트에 대한 스크립트 분석 과정을 거친 후 사람의 개입을 최소화하여 웹 문서를 자동으로 수집할 수 있었다.

Table. 4 The results of crawling on the homepage of the Ministry of Public Administration and Security

URLs	5,693
Number of URL with scripts	22,346
Extracted data	374,681 KB

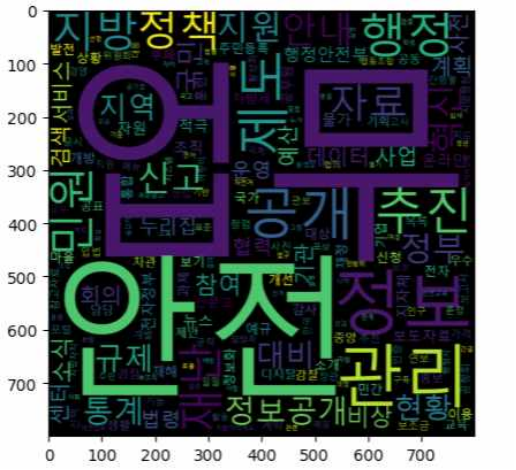


Fig. 4 The Word Cloud of crawling on the homepage of the Ministry of Public Administration and Security

Ⅴ. 결 론

행정안전부 홈페이지의 정적 주소와 스크립트 기반 주소의 비율은 1:4로 많은 정보를 심층 웹에서 제공하고 있음을 보였다. 이처럼 많은 양의 정보를 제공한다고 하더라도 심층 웹에서 제공되는 정보는 검색엔진을 통해 정보가 제공되지 않는다. 키워드 기반으로 이러한 정보를 수집하기 위해서는 사이트별로 최적화된 질의어 사전이 필요할 것이다. 한편 대부분의 웹사이트는 질의어 입력에 대해 결과를 보여주는 것에 더해 링크 클릭으로 정보 대부분을 보여준다. 본 논문에서는 이러한 점에 착안하여 모든 링크에 접근 가능한 알고리즘을 제안하였다. 스크립트를 실행하기 위해서는 링크 정보를 알고 있는 것 이상의 처리 과정이 필요했지만, 심층 웹 관련 스크립트가 기술적 유사성을 가지고 있는 것을 고려할 때 범용성 있는 심층 웹 크롤러로의 발전도 가능하리라 본다.

향후 연구에서는 심층 웹 수집을 효율적으로 처리하기 위한 스크립트 오류 처리, 접속 및 대기 시간 최적화, 다양한 형태의 웹페이지의 정보를 수집하기 위한 범용성 문제들을 해결하는 방법을 연구하고자 한다.

ACKNOWLEDGEMENT

This paper was supported by Wonkwang University in 2021.

REFERENCES

[1] I. Hernández, C. R. Rivero, and D. Ruiz “Deep Web crawling: a survey,” *World Wide Web*, vol. 22, pp. 1577-1610, May 2019. DOI: 10.1007/s11280-018-0602-1.

[2] White Paper: The Deep Web: Surfacing Hidden Value [Internet]. Available: https://quod.lib.umich.edu/j/jep/3336451.0007.104?view=text;rgn=main.

[3] M. A. Kausar, V. S. Dhaka, and S. K. Singh, “Web Crawler: A Review,” *International Journal of Computer Applications*, vol. 63, pp. 31-36, Feb. 2013. DOI: 10.5120/10440-5125.

[4] B. Ahuja, A. Anuradha, and A. Ashish, “Hidden Web Data Extraction Tools,” *International Journal of Computer Applications*, vol. 82, no. 15, pp. 9-15, Nov. 2013. DOI: 10.5120/14238-2377.

[5] M. Álvarez, J. Raposo, A. Pan, F. Cacheda, F. Bellas, and V. Carneiro, “DeepBot: a focused crawler for accessing hidden web content,” in *Proceedings of the 3rd international workshop on Data enginering issues in E-commerce and services: In conjunction with ACM Conference on Electronic Commerce*, San Diego: CA, USA, pp. 18-25, 2007. DOI: 10.1145/1278380.1278385.

[6] S. Raghavan and H. Garcia-Molina, “Crawling the Hidden Web,” in *Proceedings of 27th International Conference on Very Large Data Bases (VLDB 2001)*, Rome, Italy, pp. 129-138, 2001.

[7] J. Edwards, K. McCurley, and J. Tomlin, “An adaptive model for optimizing performance of an incremental web crawler,” in *Proceedings of the 10th international conference on World Wide Web*, HongKong, pp.106-113, 2001. DOI: 10.1145/371920.371960.

[8] H. Oh, D. Won, C. Kim, S. Park, and Y. Kim, “Design and implementation of crawling algorithm to collect deep web information for web archiving,” *Data Technologies and Applications*, vol. 52, no. 2, pp. 266-277, Mar. 2018. DOI: 10.1108/DTA-07-2017-0053.

[9] Beautiful Soup [Internet]. Available: https://www.crummy.com/software/BeautifulSoup/bs4/doc/.

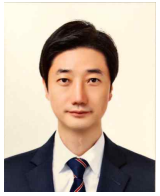
[10] ChromeDriver [Internet]. Available:
https://chromedriver.chromium.org.
[11] MongoDB [Internet]. Available:
https://www.mongodb.com/.
[12] WordCloud [Internet]. Available:
https://pypi.org/project/wordcloud/.

강윤정(Yun-Jeong Kang)

2006년 8월 전북대학교 전산통계학과 이학박사
2022년 현재 원광대학교 교양교육원 조교수
※관심분야 : 지식표현, 상황인식, 인공지능

이민혜(Min-Hye Lee)

2010. 2. : 군산대학교 컴퓨터정보공학과 공학사
2012. 8. : 원광대학교 전자공학과 공학석사
2018. 2. : 원광대학교 전자공학과 공학박사
2020. 3. ~ 현재 : 원광대학교 교양교육원 조교수
※관심분야 : 의공학, 영상처리, 인공지능



원동현(Dong-Hyun Won)

2003년 2월 전북대학교 컴퓨터과학과 공학사
2006년 2월 전북대학교 컴퓨터공학과 공학석사
2017년 2월 전북대학교 컴퓨터공학과 공학박사
2022년 현재 원광대학교 교양교육원 조교수
※관심분야 : 클라우드컴퓨팅, 인공지능, SW교육