Journal of The Korea Society of Computer and Information
Vol. 21 No. 4, pp. 55-62, April 2016

www.ksci.re.kr
http://dx.doi.org/10.9708/jksci.2016.21.4.055

Design and Adaptation for Internet News Data Extraction Middleware(INDEM) System

Bok-Keun Sun*

Abstract

In this paper, we propose the INDEM(Internet News Data Extraction Middleware) system for the removal of the unnecessary data in internet news. Although data on the internet can be used in various fields such as source of data of IR(Information Retrieval), Data mining and knowledge information service, it contains a lot of unnecessary information. The removal of the unnecessary data is a problem to be solved prior to the study of the knowledge-based information service that is based on the data of the web page. The INDEM system parses html and explores the XPath, and it is to perform the analysis. The user simply utilize INDEM by implementing an abstract class that provides INDEM, and can obtain the analysis information. INDEM System through this process delivers the analysis information including the main contents of news site to the users. In this paper, the INDEM system was adapted in a stand-alone and web service system and it was evaluated on the basis of 16 news site. As a result, performance of the INDEM system is affected in html source data size and complexity of used html grammar than the main news data size.

▸ Keyword : Information Extraction, Web news page, Middleware, XPath Grouping, Data Mining

I. Introduction

웹 브라우저를 통해 접근이 가능한 인터넷 데이터의 대다수는 HTML 문서로 이루어져있다. 인터넷 정보 검색 및 추출을 위해서는 HTML 문서의 분석이 선행 되어야 하며, 자동화 된 정보의 검색 및 추출을 위해서는 여러 단계의 과정을 거쳐야 한다. 현재 주목 받고 있는 정보기술 분야의 주요 관심은 빅 데이터, 기계학습 및 인공지능 등의 기술을 활용하여 산재한 데이터를 자동으로 처리한 후, 의미 있고 유용한 정보를 구조화하여 사용자에게 제공하는 것이다.

전통적인 정보검색 및 추출은 자연어 처리기술이 활용되며, 인터넷 데이터의 경우 일반적으로 기계학습, 패턴인식, 인터넷 페이지의 구조분석 기술 등이 활용된다. 인터넷 뉴스, 특정 분야의 게시판 데이터 등 템플릿의 활용을 통해 서버 측에서 가공된 데이터의 경우 정보검색 기술을 활용한 검색 및 추출 전에 불필요한 정보의 제거 과정이 반드시 필요하며, 이를 위해서 인터넷 페이지의 구조분석을 통한 본문의 추출방법이 많이 활용되고 있다.

템플릿을 활용 한 대부분의 웹 문서는 광고, 탐색 버튼, 각종 배너, 본문과 관련 된 페이지 링크정보, 불필요한 이미지, 애니메이션 등 해당 웹 문서의 정보와 직접 관련이 없는 불필요한 데이터를 함께 포함하고 있는 경우가 대다수이다. 이러한 요소는 자동화를 통한 인터넷 데이터의 정보검색 및 추출의 방해요소로 존재하고 있으나, 사라질 가능성은 없는 데이터라 볼 수 있다. 페이지 내 본문의 내용과 상관없는 데이터를 제거하고 본문만 추출하기 위해 관련연구와 같은 다양한 접근이 시도되고 있다.

인터넷 뉴스 데이터의 본문을 추출하기 위해 본 논문에서 구현한 INDEM 시스템은 자바를 기반으로 구현되었으며, 표준화 된 API 및 데이터 모델의 제공과 추상화를 통해 다양한 시스템에서

• First Author: Bok-Keun Sun, Corresponding Author: Bok-Keun Sun
*Bok-Keun Sun(bksun@hoseo.edu), Dept. of Computer Engineering, Hoseo University
• Received: 2016. 02. 25, Revised: 2016. 03. 10, Accepted: 2016. 03. 22.

적용이 용이하도록 설계되었다. 사용자는 추상화의 구현을 통해 필요한 뉴스 데이터의 본문을 추출 할 수 있으며, 데이터 모델을 통해 추출 된 본문 데이터 및 분석 정보를 획득할 수 있다.

2장에서는 웹페이지의 구조분석과 관련한 연구, 3장에서 INDEM의 구성요소별 구조 및 구현원리에 대해 논하며, 4장에서 설계 및 구현원리에 대해, 5장과 6장에서 적용 및 성능평가와 미구현 부분의 향후 설계 방향에 대해 논하도록 한다.

II. Preliminaries

1. Structure of a typical internet news page

그림 1은 뉴스 페이지의 한 예를 보여주며, 정보처리 시스템의 관점에서 볼 때, 해당 페이지의 요소 중 뉴스 기사의 본문 이외에는 모두 불필요한 데이터라고 할 수 있다. 뉴스 서비스를 제공하는 사이트마다 사용되는 HTML, CSS 및 자바스크립트 등 웹 구성요소는 다르지만 사용자에게 제공되는 화면은 그림 1과 유사한 형태를 나타낸다.

Fig. 1. Web News Page Sample(ex-media.daum.net)

한국에서 제공하는 10개 뉴스 사이트를 대상으로 뉴스 페이지의 전체 텍스트 대비 본문 콘텐츠의 텍스트 길이를 조사한 결과 총 텍스트 데이터 중 10.6%~57.8%가 실제 뉴스 본문이며, 나머지 텍스트는 모두 기사와 상관없는 링크, 댓글이나 광고에 활용되는 텍스트임을 알 수 있다[1].

그림 1과 같은 뉴스 본문을 포함한 페이지는 템플릿을 활용하여 작성되기 때문에 추출이 필요한 뉴스 도메인에서 사용하는 템플릿의 구조를 파악할 경우 그 도메인의 뉴스 본문을 추출할 수 있게 된다. 그러나 뉴스 도메인마다 사용되는 템플릿의 구조는 모두 다르며, 이러한 구조를 파악하기 위해 [2][3]과 같은 연구가 이루어지고 있으나, 계산에 소모되는 비용이 크며, 템플릿 구조가 바뀔 때마다 새로운 비용이 발생하는 단점을 가지고 있다.

2. Related works

정보검색 서비스 및 응용분야, 지식 정보서비스를 위한 원천 데이터로써 웹 문서를 사용하기 위해서는 전처리 과정을 통해 웹 데이터 추출이 이루어져야 하며, 정확한 데이터의 추출을 위해 다양한 관점에서 연구가 이루어지고 있다.

[4]는 HTML 트리를 만들고 그 중 블록 속성을 가진 태그의 구조를 재정의 한 후 이의 분석을 통해 중국어 웹페이지의 본문 콘텐츠를 추출한다. 분석과정에서 사용되는 파라미터로는 텍스트의 길이, 구두점의 개수, 하이퍼링크의 수 등이 활용되며, 임계값의 설정을 통해 해당 블록 안의 콘텐츠가 메인 데이터인지 노이즈 데이터인지 확인하게 된다. FODEX[5]는 본문 및 관련된 쓰레드의 구조를 파악하여 웹 뉴스 포럼의 데이터 추출을 진행하였으며, Clearly[6], Readability[7]은 Chrome 및 안드로이드 확장 프로그램으로 <body> 태그 이하 모든 노드를 검사하면서 본문의 후보가 될 노드의 점수를 산출하는 방식으로 본문을 선정하게 된다.

템플릿 화 된 웹 페이지의 본문 추출 방법 중 DOM (Documents Object Model)을 활용한 연구가 활발하게 이루어지고 있다. [2]는 템플릿을 활용하여 웹 포럼 콘텐츠를 추출하는 연구를 진행하고 있다. 웹 페이지의 구조변화에도 대응이 가능하면서 효율적으로 정보추출이 가능하도록 DOM 트리를 활용하여 데이터를 추출하는 기법을 사용한다. [8]은 HTML 파서를 활용해 DOM 트리를 구성하고 탐색하면서 필터링 알고리즘을 활용하여 해당 페이지의 메인 콘텐츠를 추출한다. DOM을 활용한 10개의 연구를 측정연구 한 [9]의 결과를 요약하면, 시스템의 복잡도, 내부 파라미터의 재조정 문제, 다량의 링크 등 HTML 페이지의 구성에 따라 달라지는 성능문제 등이 제기되고 있다.

DANA[10]은 영어를 사용하지 않는 국가의 웹 콘텐츠는 영어가 아니라는 점에 착안하여 인코딩 정보를 활용하여 영어로 되어있지 않은 사이트의 본문 콘텐츠를 추출하는 시스템으로 ASCII정보와 non-ASCII정보의 분류 후 연산을 통해 해당 페이지의 주 데이터를 추출 한다.

[1]은 뉴스 본문과 상관없는 불필요한 데이터를 제거하고 순수하게 본문에 해당하는 데이터를 추출하기 위하여 입력으로 들어온 뉴스 페이지를 XPath를 활용한 트리 형태의 자료구조로 구성 한 후, 분석을 통해 해당페이지의 본문을 자동으로 추출할 수 있도록 구성하였다.

본 논문에서는 java 응용프로그램으로 개발 된 [1]의 XPath Extractor(XTractor)를 미들웨어 시스템에 포함하여 INDEM을 설계하고 구현하였으며, 자바 어플리케이션 및 웹 서비스를 통해 INDEM 시스템의 적용 및 평가를 수행하였다.

III. Extraction Logic and Algorithm

1. Extraction logic

1.1 XPath

XPath는 XML 문서의 요소데이터와 속성데이터의 탐색을 위해 사용되어지며, W3C의 XSLT 표준의 핵심 요소이다 [11-12]. XPath는 XML 문서의 노드 또는 노드집합의 선택을 위한 경로의 표현방법으로 사용될 수 있으며, 문자열이나 숫자 등과 관련된 함수를 제공한다. 또한 XSLT, XPointer, XQuery 등 XML 문서를 다루는 다른 표준에서도 활용된다.

XPath는 XML문서의 특정한 요소나 속성까지 도달하기 위한 경로를 html/div/div/ul/li...와 같은 트리형태의 계층 구조를 이용해서 표현하며, 본 논문에서는 뉴스데이터의 본문이 포함된 HTML 파일의 파싱을 통해 <HTML> tag를 root로 하는 요소(element)트리를 만들면서 트리의 각 요소마다 XPath를 포함하도록 구성하였다.

1.2 XPath selection logic

XPath를 가지고 있는 전체 요소트리 중 텍스트 데이터(pcdata-printable character data)를 포함한 요소 수를 N으로 보았을 때, 수식(1)의 XPath는 N개의 요소를 가진 중복집합(multiset)으로 표현된다. 입력 데이터에서 연속되는 동일한 XPath(ID)의 중복도(multiplicity, C) 계산을 통해 요소집합으로 구분하였다. 텍스트가 포함된 요소를 수식(1)과 같이 중복집합으로 구분할 경우, n개의 요소블록 그룹을 검출할 수 있다.

$$\begin{cases} \begin{cases} XPATH = \{ID_N, \{ID_1, C_1\}, \{ID_2, C_2\}, ..., \{ID_n, C_n\}\} \\ ID : XPATH \rightarrow N \geq 1 = \{1,2,3,...N\} \end{cases} \\ N = \sum_{i=1}^{n} C_i \end{cases} \quad (1)$$

검출 된 요소블록(ID1-IDn)은 모두 뉴스 본문을 포함하는 XPath의 가능성을 가지고 있다고 볼 수 있으며, 이중 하나를 선정하여 뉴스 본문을 가진 요소블록으로 판정하게 된다. 판정을 위해 수식(2)와 같이 각 ID가 가지고 있는 문자의 길이를 모두 합산하여 가장 길이가 긴 데이터를 가지고 있는 ID를 뉴스 본문을 가진 XPath로 선정하게 된다.

$$\begin{cases} Length\,of\ ID_n = \sum_{i=1}^{C_n} textLength\,of\ ID_n \\ Candidate\ ID = \max(ID_1, ID_2, ...ID_n) \end{cases} \quad (2)$$

2. Main contents extraction algorithm

뉴스 본문 검출은 네트워크상의 HTML 문서를 입력으로 하며, 중간에 요소트리, XPath 그룹화, 본문 XPath의 3가지 중간 데이터를 산출한다.

2.1 Element and element tree

입력으로 들어온 HTML 문서는 파서를 통해 요소 트리로 구성된다. 파서는 HTML문서의 태그별 토큰화(tokenization)를 통한 요소 생성과 트리구조화를 반복하면서 요소트리를 생성하게 된다. 그림 2는 XTractor 내부의 파싱관련 클래스 및 요소트리에 해당하는 Element 클래스 다이어그램의 요약을 나타낸다.

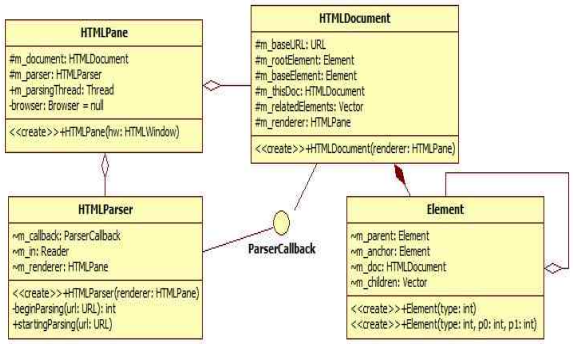

Fig. 2. A Summary of the Class Associated with the HTML Parsing

HTMLParser 클래스는 입력된 HTML파일의 태그를 토큰화하여 ParserCallback 인터페이스에 전달한다. ParserCallback 인터페이스는 HTML 태그를 root로 하는 Element의 생성을 시작으로 하위 Element를 생성하여 트리구조화를 수행하며, 파싱이 끝나게 되면 요소트리가 생성된다. 그림 3은 요소트리 중 pcdata를 가진 요소트리의 개념도를 나타낸다.

요소트리 생성은 문서의 모든 태그에 대해 이루어지며, 파싱이 끝나게 되면 요소트리 탐색을 통해 그룹화와 본문후보의 XPath를 선택하게 된다.

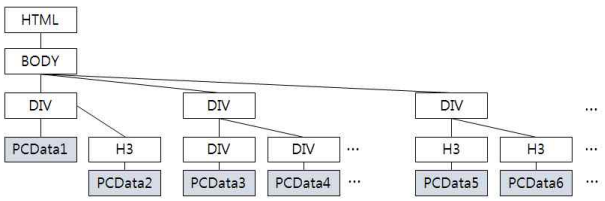

Fig. 3. Schematic View of the Element Tree with the pcdata

2.2 Element tree traverse and grouping algorithm

요소트리의 각 요소는 고유의 XPath를 가지고 있으며, pcdata를 가지고 있는 요소트리만 선별하여 연속적으로 발견되는 동일 XPath를 가진 요소별로 그룹화가 가능하다.

그림 3에 나타난 pcdata의 XPath는 다음과 같다.

- PCData1 : HTML/BODY/DIV/
- PCData2 : HTML/BODY/DIV/H3
- PCData3 : HTML/BODY/DIV/DIV

- PCData4 : HTML/BODY/DIV/DIV
- PCData5 : HTML/BODY/DIV/H3
- PCData6 : HTML/BODY/DIV/H3

생성된 요소트리 탐색은 root 요소부터 시작하여 pcdata를 포함하는 최하위 노드 요소까지 깊이우선 탐색(depth-first search)하면서 XPath의 그룹화를 진행한다. 그림 4는 그룹화 알고리즘의 슈도코드를 나타내며, 그룹화 후에는 그림 3의 요소트리가 그림 5와 같이 그룹화 된다.

```
Void AnalizeFunc(Node node)
  IF PCDATA Node
    IF Node==new Node
      Save Previous Node group length
    ELSE continue to add length
  END IF
  ELSE
    WHILE(child node is Not NULL)
      Get child node
      AnalizeFunc recursive call
    END WHILE
  END ELSE
```

Fig. 4. Pseudo code of Grouping Algorithm

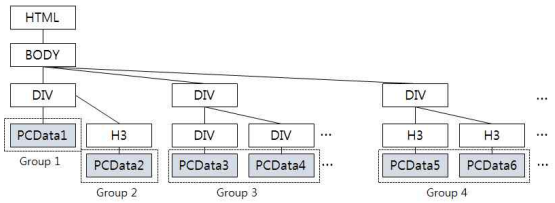

Fig. 5. Schematic View of the pcdata node grouping

요소트리 탐색 후 요소트리의 그룹화를 통해 HTML 페이지의 모든 pcdata는 XPath 그룹에 매핑되고, 생성된 그룹 중 pcdata의 길이가 가장 긴 XPath를 선정하여 해당 pcdata를 뉴스 본문으로 선정하게 된다.

IV. Design of INDEM System

INDEM 시스템은 인터넷 데이터의 접속, 데이터 다운로드, 불필요한 데이터의 제거, 본문 데이터의 추출과 관련 한 모든 구현을 추상화하여 제공한다. 그림 6은 INDEM 미들웨어의 구조를 나타낸다.

미들웨어 사용자는 Notification Interface에 추상화 되어있는 INDEM Wrapper를 상속함으로써 웹 뉴스데이터를 추출할 수 있다. Wrapper는 init(), dispose(), AnalyzeNews(), getWrapperName()의 4개 메서드를 추상화 하였으며, 웹 데이터의 파싱 상태 및 분석 상태 정보를 전달하기 위해 Notification Interface에 정의 된 메서드를 구현하였다.

Fig. 6. INDEM System Container Architecture

1. Control flow

INDEM을 사용하는 어플리케이션은 그림7과 같이 두 단계를 거쳐 데이터의 본문을 추출하게 된다. 먼저 Wrapper를 상속 받으면 Analyze manager의 인터페이스가 Wrapper에 등록된다. 이후 구동을 위해 init() 추상 메서드를 호출하면 Wrapper의 precall() 함수가 구동되며, 해당 함수의 동작을 통해 파싱 쓰레드가 동작하면서 데이터 추출이 이루어진다. Newtork Manager는 인터넷 연결을 통해 XPath Extractor에게 HTML 문서를 전달하며, 파싱쓰레드에서 요소트리를 만들고, XPath 데이터를 구축한다.

파싱 쓰레드가 끝나면, init()에서 초기화 된 Analyzer Manager의 분석 쓰레드가 이어서 동작하며 3장에서 논한 알고리즘에 따라 본문이 포함 된 XPath를 결정한다. 결정과 함께 분석이 종료되고, 분석 데이터를 포함하는 NewsDataSet을 사용자에게 전달한다.

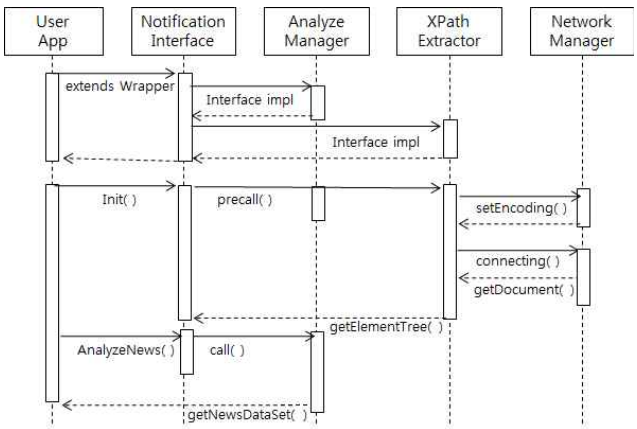

Fig. 7. Control flow for a adaptation processing

2. Processed data model

INDEM 시스템의 데이터 생성 내용은 그림 8과 같다. 네트워크로부터 들어온 HTML 데이터는 Network Manager에서 버퍼에 저장되고, XPath Extractor에서 XPath 리스트를 포함

한 요소트리로 만들어진다. Analyze Manager는 수식 1, 2와 그림 4에서 제시한 알고리즘에 따라 요소트리를 분석하여 본문 XPath를 추출하게 된다. 본문의 후보가 선정되면, NewsDataSet을 만들어 사용자에게 전달한다.

사용자가 상속받은 Wrapper의 NewsDataSet에는 분석과정에서 도출되는 모든 중간데이터가 함께 포함되어 사용자의 목적에 맞게 활용 및 재사용 할 수 있도록 제공된다.

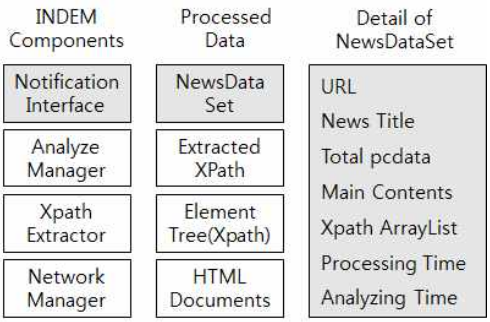

Fig. 8. processed data model of INDEM

V. Adaptation and Evaluation

INDEM 시스템의 적용은 JDK SE1.8.0 가상머신 위에서의 단일 시스템과 아파치 톰캣 서버를 통한 웹 서비스 시스템에서 이루어졌으며, 시스템의 평가를 위해 HTML 데이터의 전체 크기, XPath의 개수, pcdata의 크기에 따른 본문 추출 시간을 측정하였다. 다음 절부터 각각의 적용과 분석내용을 기술한다.

1. Adaptation of INDEM system

그림 9는 단일 시스템 평가를 위해 Notification Interface의 Wrapper클래스를 상속한 testStandAlone 클래스의 소스코드이다.

```
package testINDE;

import wrapper.INDEGeneralWrapper;
public class testStandAlone extends INDEGeneralWrapper{
    private static int threadcounter = 0;
    public testStandAlone(){
        super();
    }
    public FutureTask<NewsDataSet> AnalyzeNews(){
        FutureTask<NewsDataSet> ds = new FutureTask<NewsDataSet>(this);
        new Thread(ds).run();
        return ds;
    }
    public  String getWrapperName(){
        return "test Wrapper";
    }
    public  void dispose(){
        threadcounter--;
        System.gc();
    }
    public  void init(String url, String charset){
        threadcounter++;
        setName("test Wrapper"+threadcounter);
        url_str = url;
        this.charset = charset;
        preCall();
    }
}
```

Fig. 9. Example source code for adaptation of stand alone system

해당 클래스는 Wrapper 클래스의 4개 추상 메서드를 구현하였으며, 적용 예를 보이기 위해 GUI를 적용한 실행 화면은 그림 10과 같다. 그림 10은 그림 11의 뉴스 데이터 본문을 추출한 화면이며, NewsDataSet의 데이터를 GUI 형태로 나타낼 수 있도록 적용하였다.

Fig. 10. Result screen of stand alone system

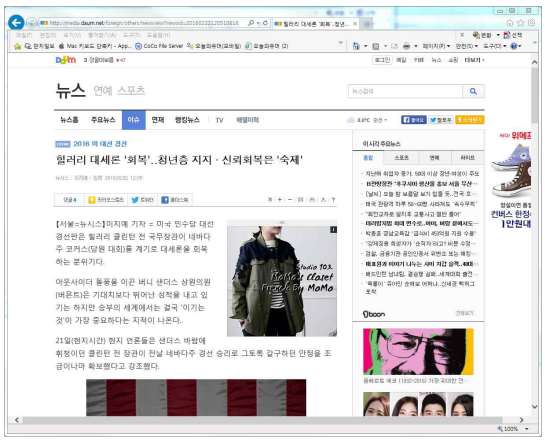

Fig. 11. Source screen of input news site (example of media.daum.net news page)

두 번째 적용은 미들웨어 시스템을 WAR 포맷으로 변환하여 아파치 톰캣 서버와 연동 후, 네트워크를 통하여 그림 11과 동일한 페이지에 접속한 후 본문을 추출하였다. 그림 12와 같이 간단한 JSP 파일을 작성하여 적용 하였으며, 결과는 그림 14와 같다. JSP 파일에서 호출되는 ExtractorTest 서블릿 또한 그림 9의 클래스와 같이 Wrapper를 상속하여 작성되었다.

```
<body>
    <form method="post" action="ExtractorTest">
    url 입력
    <input type="text" name="inURL"><p>
    <input type="submit" value = "extract">
    </form>
</body>
```

Fig. 12. Example source code for adaptation of web service

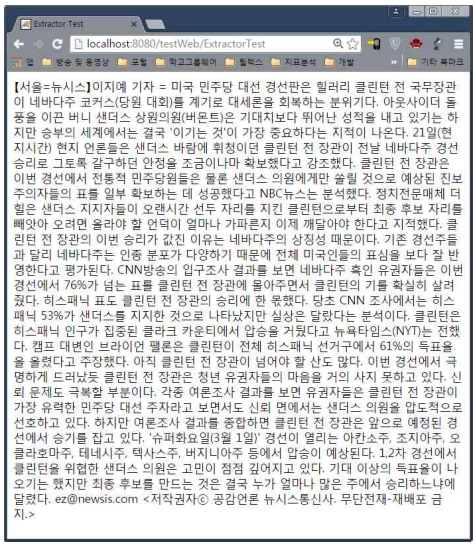

Fig. 13. Result screen of web service

2. Evaluation of INDEM system

INDEM 시스템의 평가 방법은 본문 추출에 대한 정확도 평가와 본문을 추출하는데 소요되는 시간으로 나뉠 수 있다. 본 논문에서 구현한 시스템은 XPath를 활용한 문서추출 시스템인 XTractor[1]를 기반으로 구축되었으며, 해당 논문에 본문(N)에 대한 인식률(R)과 정확률(P)를 측정한 결과가 제시되어있다.

$$\begin{cases} \text{인식률}(R) = \dfrac{N-r}{N} \times 100 \\ \text{정확률}(P) = \dfrac{M}{M+G} \times 100 \end{cases} \quad (3)$$

인식률과 정확률은 수식 3과 같다. 인식률로 평가 시스템이 페이지의 본문에 해당하는 부분을 정확히 검출했는지 평가한다. 본문을 찾지 못하거나, 본문과 전혀 다른 내용을 본문으로 제시할 경우(r) 시스템이 인식하지 못한 것으로 평가하였다. 정확률은 평가 시스템이 제시 한 본문이 실제 본문(M) 외에 불필요한 데이터(G)를 얼마나 포함하고 있는지 평가한다. 정확률이 높을수록 본문 외의 불필요한 데이터가 적다는 것을 나타낸다. XTractor의 평균 인식률은 97.9%이며, 정확률은 93.9%로 측정되었다.

본문 추출에 소요되는 시간은 크게 네트워크를 통해 HTML 데이터를 다운받는 시간, XPath를 검출하고 본문 후보 XPath를 검사하는 시간, NewsDataSet 데이터를 생성하여 사용자에게 전달하는 시간으로 나뉜다. HTML 데이터를 다운받는 시간은 네트워크 상황에 따라 상이하므로 평가에서 제외하였다.

시스템의 테스트 과정에서 XPath검출에 대부분의 시간이 소요되며, 본문후보 추출과 NewsDataSet 생성에 소요되는 시간은 밀리초 이하의 시간이 소요되는 것으로 측정된 바, 데이터 처리시간으로 통합하여 계산하였다.

측정에 사용된 시스템은 윈도우즈7 64bit 운영체제, 인텔 i5 3GHz CPU, 8GB RAM 성능의 PC이며, 국내외 16개 뉴스 사이트 50개의 문서를 대상으로 시스템의 데이터 처리시간에 대한 평가를 진행하였다.

문서 당 HTML 데이터의 전체 크기, 전체 pcdata의 크기, 본문으로 선정된 pcdata의 크기에 따른 본문 추출 시간을 각각 측정하였으며, 측정 결과는 각각 그림 14, 그림 15, 그림 16과 같다.

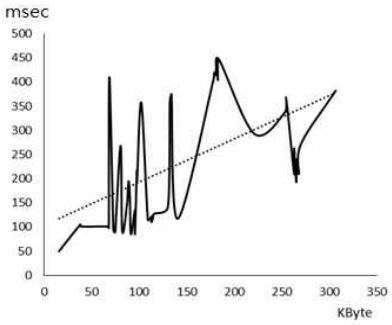

Fig. 14. HTML size and processing time ratio

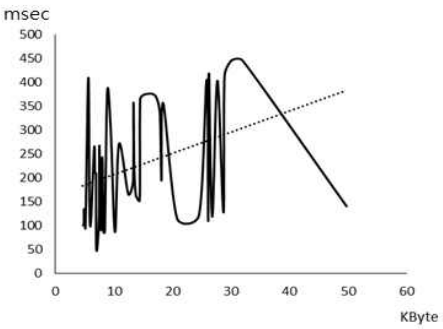

Fig. 15. Total pcdata size and processing time ratio

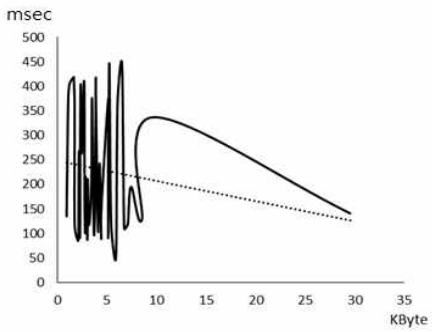

Fig. 16. Main contents size and processing time ratio

그림 14와 그림 15는 HTML 데이터 크기와 HTML내 pcdata 크기에 따른 본문 추출 시간을 분석한 그래프이다. 사용된 입력 데이터의 크기와 데이터 추출에 소요되는 시간관계는 완전 비례관계는 아니지만 추세선을 볼 때, 완만한 경사의 정비례 분포를 보인다. 그러나 추출 된 본문 뉴스 데이터의 크기에 따른 본문 추출 시간을 분석한 그림 3을 볼 때, 본문 데이터의 크기와 본문 추출 시간은 크게 상관관계가 없는 것으로 나타난다.

결론적으로 뉴스 페이지가 가지고 있는 데이터 중 본문 뉴스 데이터의 크기보다 HTML 데이터의 크기 및 사용된 HTML의 복잡도에 따라 본문 추출 성능이 달라질 수 있음을 확인할 수 있었다. 또한, 동일 템플릿을 사용하는 뉴스 제공 사이트의 서로 다른 뉴스 페이지 역시 본문의 크기와 상관없이 비슷한 본문 추출 성능을 나타냄을 확인하였다.

VI. Conclusions

본 연구를 통해 인터넷 뉴스 데이터의 본문을 추출하고자하는 사용자의 시스템에 적용할 수 있는 미들웨어 시스템을 구현하고 평가하였다. 사용자가 INDEM 시스템의 Notification Interface에서 제공하는 Wrapper를 상속하고, 추상 클래스를 구현함으로써 원하는 결과를 얻을 수 있도록 구성하였으며, 시스템의 핵심이 되는 XPath 처리모듈은 [1]을 활용한 XPath Extractor를 사용했다. 단일 시스템과 네트워크 시스템 상에서 미들웨어를 적용한 예시를 구현하였으며, 평가결과 HTML 데이터의 크기 및 사용된 HTML의 복잡도에 따라 본문 추출 성능이 달라질 수 있음을 확인할 수 있었다.

완전한 미들웨어로서의 동작을 위해서는 INDEM이 별도로 구동되고, 본문 추출을 원하는 사용자에게는 XML과 같은 표준화 된 데이터 교환 정책과 클래스 동적 할당이 제공되어야 한다. 또한 빅 데이터 처리 등에 활용하기 위해서는 형태소 분석 모듈의 추가 및 DB화, 랭킹 등의 문서요약 알고리즘 적용이 필요하며, 향후 과제로써 해당 내용을 본 시스템 접목해보고자 한다.

REFERENCES

[1] B. K. Sun, “A Study of Main Contents Extraction from Web News Pages based on XPath Analysis”, Journal of The Korea Society of Computer and Information, Vol. 20, No. 7, pp. 1-7, July 2015.

[2] J. Si, W. Wang, “A Template-based forum posts content extraction method”, International Conference on ICECE, pp.38-41, 2011.

[3] R. Gunasundari, S. Karthikeyan, “A Study of content extraction from web pages based on links”, International Journal of Data Mining & Knowledge management Process(IJDKP) vol.2, No.3, May 2012.

[4] B. Zhou, C. Wang, Q. Su, “Chinese web page content extraction based on page content analysis”, Journal of Computational Information Systems vol.5, No.6, pp.1861-1871, Dec 2009.

[5] S.Pretzsch, K.Muthmann, A.Schill, “FODEX-Towards generic data extraction from web forums”, 26th International conference on advanced information networking and applications workshops, pp.821-826, 2012.

[6] Clearly, *https://chrome.google.com/webstore/ detail /clearly/iooicodkiihhpojmmeghjclgihfjdjhj*

[7] Readability, https://www.readability.com/

[8] S.Gupta, G. Kaiser, D. Neistadt, and P. GS.Gupta, G. Kaiser, D. Neistadt, and P. Grimm, "DOM-based content extraction of HTML documents", in WWW '03: Proceedings of the 12th International Conference on WWW, ACM, pp.207-214, 2003.

[9] N. Negm, P. Elkafrawy, A.B. Salem, “A Survey of Web Information Extraction Tools”, International Journal of Computer Applications, Vol. 43, No. 7, pp.19-27, April 2012.

[10] H. Mohammadzadeh, T. Gottron, F. Schweiggert, G. Nakhaeiza, “A Fast and accurate approach for main content extraction based on character encoding”, 22nd International workshop on database and expert systems applications, pp.167-171. 2011.

[11] SY. Oh, “X2RD: Storing and Quering XML Data Using XPath to Relational Database”, Journal of The Korea Society of Computer and Information, Vol. 14, No. 3, pp. 57-64, March 2009.

[12] XPath, http://www.w3.org/TR/xpath/

Author

Bok Keun Sun received the B.S., M.S. and Ph.D. degrees in Computer Engineering from Hoseo University, Korea, in 1999, 2001 and 2006, respectively. Dr. Sun joined the faculty of the Department of Computer Engineering at Hoseo University, ChungNam, Korea, in 2008. He is currently a Professor in the Department of Computer Engineering, Hoseo University. He is interested in data mining, embedded system and IoT computing