JDCS
디지털콘텐츠학회논문지
Journal of Digital Contents Society
Vol. 24, No. 4, pp. 691-699, Apr. 2023

Check for updates

HTML 본문 추출을 위한 새로운 시각적 Feature

정 근 성[1] · 차 재 혁[2*]
[1]한양대학교 컴퓨터소프트웨어학과 박사과정
[2*]한양대학교 컴퓨터소프트웨어학과 교수

New Visual Features for HTML Main Content Extraction

Geunseong Jung[1] · Jaehyuk Cha[2*]

[1]Doctoral Student, Department of Computer Science, Hanyang University, Seoul 04763, Korea
[2*]Professor, Department of Computer Science, Hanyang University, Seoul 04763, Korea

[요 약]

HTML 본문 추출이란 웹페이지의 본문 영역과 그 내용을 파악하는 기술이다. 기존 기술들이 본문 구별을 위해 사용하는 feature 는 주로 HTML 노드의 태그로 구성한 구조적 feature 이거나 노드가 포함하는 텍스트의 통계값으로 이루어진 텍스트 feature 이다. 그러나 이 feature 들은 웹페이지 템플릿의 유행, 언어, 지역 등에 의존적이다. 따라서 이 feature 들을 활용한 알고리즘이나 모델은 웹페이지의 언어나 환경으로 인한 성능 편차가 발생할 수 있다. 따라서 본 논문에서는 다국어 웹페이지에 대한 HTML 본문 추출 성능 저하를 최소화한 새로운 시각적 feature 들을 제안한다. 이 feature 들은 브라우저에 렌더링 된 HTML 노드의 결과의 속성에 기원하며, 언어나 지역의 영향이 상대적으로 적다. 본 논문에서는 Google TabNet 심층 신경망 아키텍처를 활용하여 기존의 구조적, 텍스트 feature 만을 학습한 신경망 모델 및 기존 feature 에 새롭게 제시한 시각적 feature 을 추가한 모델을 각각 학습하고 본문 추출 성능을 비교하여 본 논문에서 제시한 시각적 feature 의 성능 개선 효과를 입증하였다.

[Abstract]

Hypertext markup language (HTML) main content extraction is a technology that identifies the body and contents of an article from web pages. Traditional technologies use structural features, such as the tag structure of the HTML node and text features based on statistical properties. However, because these features depend on web development trends, language, and the region of the webpage, the performance of algorithms or models based on these features can vary. Therefore, in this study, we propose a novel visual feature to prevent the degradation of HTML body extraction performance on multilingual web pages. The feature is based on the results of HTML node attributes rendered in the browser; therefore, the influence of the language or region is relatively small. The Google TabNet deep neural network architecture was used to learn the neural network model based on only structural and text features, and subsequently another model with the newly introduced visual feature along with the structural and text features was trained. A comparison of the body extraction performance of the two models demonstrates the performance improvement provided by visual features in this study.

색인어 : 주요 콘텐츠 추출, 웹페이지, 웹 콘텐츠 추출, 신경망 모델, Google TabNet

Keyword : Main content extraction, Webpage, Web content extraction, Deep neural net model, Google TabNet

http://dx.doi.org/10.9728/dcs.2023.24.4.691

Received 23 February 2023; **Revised** 23 March 2023
Accepted 03 April 2023

***Corresponding Author; Jaehyuk Cha**

Tel: +82-2-2220-4458
E-mail: chajh@hanyang.ac.kr

Ⅰ. 서 론

웹 기술의 발전으로 인해 웹페이지는 보다 다양한 콘텐츠를 표시할 수 있게 되었다. 때문에 최근의 웹페이지는 텍스트뿐만 아니라 이미지, 동영상 등을 포함하는 풍부한 데이터 소스로서 여겨진다[1],[2]. 정보 검색[3],[4], 자연어 처리[5],[6], HCI[7], 마케팅[8],[9] 등 수많은 분야에서 웹페이지를 주요 데이터 소스로서 다루고 있다. 그러나 현대 웹페이지는 데이터 소스로 적합한 본문 정보 외에도 광고, 메뉴, 댓글, 입력 폼 등의 부차적인 기능과 역할을 수행하는 영역들이 혼재되어 있는, 이른바 더러운 데이터 (Dirty Data) 로 유용한 데이터만을 추출하는 사전 작업이 필수불가결하다.

웹페이지의 주요 콘텐츠 추출(main content extraction) 기술은 광고 등의 목적과 관계없는 부수적인 영역을 제거하고 본문 영역만을 추출하는 기술로, 웹 크롤링, 데이터 마이닝, 텍스트 분석 등 많은 분야에서 활용하고 있다[1],[10]. 모든 웹페이지가 본문 영역을 명시하지 않고 있기 때문에 주요 콘텐츠 추출 기술은 웹페이지 안에서 어떤 영역이 주요 콘텐츠를 포함하는지 판단하는 알고리즘이나 모델이다.

전통적인 추출 기술들이 주로 사용하는 feature는 HTML의 구조적 특성에 기원한 구조적 feature, 또는 웹페이지에 존재하는 텍스트가 가진 특성에 기원한 텍스트 feature 이다. 구조적 feature는 브라우저가 웹페이지를 렌더링하기 위해 HTML 노드들을 구조화하는 과정에서 부여되는 성질을 이용하는 것으로 HTML 태그의 특성, 노드의 관계 등을 사용한다[11]. 한편, 텍스트 feature를 사용하는 기술들은 불용어(stop words) 나 특정 단어의 포함 정도, 단어 수, 단어 빈도 등의 텍스트 통계로 본문 영역을 추출한다[4],[10],[12]. 여전히 최신 추출 기술에서도 두 방식의 feature들을 사용하고 중이며, 이는 본문 영역이 가진 구조나 텍스트적 특성이 다른 영역과 유의미하게 다르다는 것을 시사한다.

그러나 이 두 방식의 feature들은 웹페이지 개발 유행, 언어, 지역 등의 요소로 인해 때때로 잘 작동하지 않을 수 있다. 예를 들어, 구조적 feature의 경우, 웹 기술의 발달로 인한 신규 HTML의 태그 추가로 다른 코드로 겉보기에 동일한 코드가 존재할 수 있으며, 블로그 플랫폼이나 포털 사이트의 업데이트 등으로 인해 같은 URL의 동일한 콘텐츠의 웹페이지가 겉모습만 달라질 수 있다. 한편, 텍스트 feature의 경우, 언어의 영향으로 인해 특정 언어를 대상으로 한 알고리즘이나 모델이 의도대로 작동하지 않을 수 있다. 가령, 영문 텍스트의 단어 수는 띄어쓰기로 셀 수 있지만, 일본어나 중국어처럼 띄어쓰기를 사용하지 않거나, 한국어처럼 별도의 띄어쓰기 규칙이 있는 경우 동일한 단어 수 측정 알고리즘을 적용할 수 없다. 따라서 텍스트 feature를 활용하기 위해서는 각 언어에 따른 규칙을 구성하거나 언어별로 새로운 모델을 학습하는 등의 작업이 필요하지만 현실적으로 모든 언어에 대해 개별적인 알고리즘을 적용하거나 신규 모델을 생성하는 것은 매우 어렵다.

본 논문에서는 개발 유행, 언어 등에 영향을 받지 않는, 웹페이지에 렌더링되어 사용자에게 전달되는 겉모습에 기반한 새로운 시각적 feature를 제시한다. 또한 이 feature의 효과를 파악하기 위해 Google TabNet 신경망[13]을 사용하여 기존 추출 기술들이 사용하는 구조적 및 텍스트 feature 만을 학습한 모델과 새로운 시각적 feature를 포함하여 학습한 모델 간의 성능 차이를 분석한다. 두 모델로 다양한 언어의 웹페이지로 구성된 데이터셋에서 본문 추출 실험을 진행하고 그 결과를 통해 feature 의 차이로 인한 성능 변화와 새 시각적 feature의 유용성을 보인다.

Ⅱ. 관련 연구

2-1 주요 본문 추출 위한 기존 Feature

웹페이지는 단순한 데이터를 전송하기 위한 포맷이 아닌, 사용자와의 시각적인 상호작용을 위해 제작되고 배포된다[2]. 따라서 웹사이트들은 웹페이지에서 자신들이 제공하는 다양한 정보와 서비스를 제공하기 위해 한 화면에 여러 기능을 가진 영역을 함께 제공한다. 이에 더해, 때때로 회원 로그인, 쿠키, 광고 ID 등 개별 사용자의 정보를 활용하여 콘텐츠를 동적으로 생성하기도 한다. 즉, 하나의 웹페이지는 고정되고 정적인 콘텐츠와 모양새를 갖지 않을 수 있으며 조건에 따라 변화할 수 있다. 따라서 웹페이지를 구성하는 HTML 코드에서 본문 영역을 하나의 방법으로 정의하고 추출하는 것은 어렵다. 대신에 기존의 주요 콘텐츠 추출 기술들은 사람이 인식하는 본문 영역과 유사한 영역을 정의하고 그것을 구분할 수 있는 여러 feature를 제시하고 이를 활용한 알고리즘이나 모델들을 제시하였다.

텍스트 feature는 본문을 제외한 영역들이 짧은 문장이나 간결한 단어 위주로 이루어진 것과 달리 본문 영역이 상대적으로 긴 문장과 잘 구성된 문장으로 구성되어 있는 것에 착안했다. 가령, 논문[14]은 웹페이지를 하나의 긴 글로 치환하여 문장의 스타일이 변하는 영역을 본문 영역으로 판단했다. 그 후에도 단어 수, 단어의 빈도, 밀도, 대소문자 등을 활용하는 추출 기술이 연구되었다[10],[12].

구조적(Structural) feature는 웹페이지를 구성하는 언어인 HTML 을 웹 브라우저가 해석하는 과정과 그 결과를 활용한다. 웹 브라우저는 HTML을 해석하여 트리 구조의 문서 객체 모델(DOM; Document Object Model)을 생성함으로써 웹 페이지를 표현하고, 조작할 수 있는 형태로 구성한다. 따라서 DOM으로 웹페이지의 정보들의 계층 등 구조적 관계를 파악할 수 있다. HTML의 태그 정보와 HTML 노드의 DOM Tree 상에서의 관계 등은 웹페이지 내의 영역들이 어떤 의도로 구조화되었는지 유추할 수 있다. 가령, 수평 가로선을 의미하는 <HR> 태그가 연속된 HTML 노드 사이에 존재한다면,

그 태그를 기점으로 앞과 뒤 노드는 서로 다른 그룹이나 영역임을 파악할 수 있다. 또한 구조적 feature는 웹사이트 템플릿 등을 파악하기 유리하므로 유명 웹사이트에서 독자적으로 사용하는 태그나 구조를 미리 파악하고 적용하여 특정 사이트 전용으로 높은 성능을 얻을 수 있다[15]. 그러나 HTML을 DOM으로 해석하는 과정은 모든 브라우저가 완벽히 동일하지 않으며, 웹페이지가 제작된 시기나 웹 프레임워크 등의 유행에 따라 지배적으로 활용되는 템플릿이나 주요 콘텐츠의 구성 방법이 달라질 수 있으므로 알고리즘이나 모델이 성능이 저하될 수 있다.

시각적 feature는 웹페이지가 브라우저에서 나타나는 겉모습을 표현하기 위한 데이터에 기반한다. 보편적으로 현대적 웹페이지들은 웹 요소의 시각적 표현을 위해 Cascading Style Sheets (CSS)를 사용하므로 DOM과 CSS를 통해 웹 콘텐츠의 위치, 모양, 크기, 글꼴, 색 등에 관한 데이터를 얻을 수 있고 이를 활용하여 시각적 feature들을 정의할 수 있다. 시각적 feature를 사용하는 주요 콘텐츠 추출 기술들은 주로 웹페이지를 시각적 요소(위치, 모양 등)을 만족하는 추상화된 블록의 집합으로 취급하여 블록들을 클러스터링하거나, 분할하는 접근을 사용한다[16].

2-2 주요 기존 본문 추출 기술

각 feature들은 배타적이지 않으며 일부 추출 기술들은 여러 feature 들을 복합적으로 사용한다.

Mozilla Readabiliy.js[15]는 Mozilla Firefox 브라우저의 읽기 모드로, 전통적인 규칙 기반의 알고리즘을 사용하는 주요 콘텐츠 추출 기술이다. 이 기술은 HTML 노드의 태그 이름, 글자 수, 링크 밀도와 더불어 텍스트 패턴이나 특정 유명 사이트의 템플릿 규칙을 포함한다. 이로 인해 성능을 유지하기 위해서는 웹 기술에 발달에 맞추어 지속적인 업데이트가 필요하다는 제약이 존재하나 저명한 브라우저 개발사의 프로젝트로서 지원이 계속되고 있어 인지도가 있는 웹사이트에서 대부분 잘 작동한다.

DOM Distiller[17]는 Google Chrome의 읽기 모드로 Boilerpipe[12]에 기반한 분류기(Classifier)에 Readability.js와 유사한 규칙 기반 알고리즘을 추가하여 개선한 것이다. 글자 수를 비롯한 텍스트, 구조적 feature를 활용하는 의사 결정 트리와 서포트 벡터 머신(SVM; Support Vector Machine)으로 구성된다.

Web2Text[4]는 합성곱 신경망(CNN; Convolutional Neural Network)을 사용하는 주요 콘텐츠 추출 기술이다. 기존의 DOM 을 압축한 형태의 Collapsed DOM(CDOM)을 구성하고 CDOM 의 각 노드들이 가진 128개의 텍스트, 구조적 feature로 학습하는 것이 특징이다.

BoilerNet[10]은 순환 신경망(RNN; Recurrent Neural Networks)의 일종인 Long short term memory network (LSTM)을 활용하고 DOM Tree 상의 경로와 노드의 단어 벡터를 feature로 사용한다. Web2Text와 같이 DOM Tree 경로가 구조적 feature로, 단어 벡터가 텍스트 feature 로 사용된 것은 동일하지만 Web2Text와는 달리 사람이 직접 찾은 feature를 사용하지 않고 신경망이 두 데이터에서 적절하게 학습을 진행한다는 것이 차이점이다.

BoilerNet과 Web2Text 모두 이미지나 텍스트 데이터와 같은 비정형 데이터 학습에 유용한 심층 신경망(딥러닝) 기반을 두며, 텍스트와 구조적 feature를 사용하는 특징이 있다. 논문[16]에서 사용하는 것과 유사한 시각적 feature는 구조적 feature와 특징을 공유하더라도 여러 웹페이지를 학습할 때에는 서로 다른 노드의 대비(배경색 대비 등)와 같은 상대적인 값을 사용해야하기 때문에 비정형 데이터의 신경망에 학습시키기가 어려웠다. 이는 웹페이지의 디자인이 모두 다르기 때문에 동일한 시각적 특징을 수치화하기 어렵기 때문이다. 따라서 본 논문에서는 서로 다른 디자인을 가진 웹페이지라도 본문이 사용자가 찾기 쉬운 곳에 위치한다는 특성에 착안하여 모든 웹페이지에 적용할 수 있는 시각적 feature를 활용한다.

Ⅲ. 새로운 시각적 Feature 정의 및 본문 추출 모델 학습

3-1 시각적 Feature: 웹페이지 중심과 노드 사이의 거리

웹페이지를 제공하는 웹사이트의 가장 중요한 목적은 사용자에게 적절한 정보와 서비스를 제공하는 것이다. 그러나 사용자는 웹페이지를 보는 방법은 브라우저와 같은 기계가 웹페이지를 해석하는 방법과는 판이하다. 사용자는 웹페이지에 접속하여 구체적으로 콘텐츠를 정독하여 정보를 파악하고 자신에게 유용하는지 판단하지 않으며, 대신 한눈에 전체적인 모양을 파악하며 그 느낌으로 웹페이지에 대한 첫인상을 결정짓는다. 이 첫인상은 1초미만의 짧은 시간 내에 결정되며 이 시간 안에 웹페이지가 좋은 인상을 전하지 못하면 사용자는 그 웹페이지가 가진 정보에 관계없이 흥미를 잃을 가능성이 높다[18]. 따라서 웹페이지의 겉모습과 본문을 구성하는 방식이 다를지라도 사용자에게 시각적 매력(visual appeal)을 확보하는 것은 웹페이지 디자인의 핵심 요소이다. 즉, 웹사이트들은 웹페이지를 구축할 때 사용자에게 유용할 만한 정보를 제공하는 것뿐만 아니라 사용자가 파악하기 쉽고 편안하다고 느끼도록 웹페이지의 요소들을 구성해야한다.

본문 영역은 웹페이지의 핵심 정보를 담고 있기 때문에 눈에 잘 보이지 않는 영역에 있기보다는 사용자가 시각적으로 쉽게 찾을 수 있는 곳에 배치되는 경향이 있다. 이러한 경향으로 인해 웹페이지의 크기가 콘텐츠의 분량에 따라 다를 수 있음에도 불구하고 대다수의 본문 영역은 화면 중심을 가로지르는 선과 웹페이지 전체의 중심을 가로지르는 선 사이의

구간에 걸쳐져 있는 것으로 나타났다[19]. 따라서 각 HTML 노드에서 두 중심선과의 상대 위치는 모든 웹페이지에 적용할 수 있으면서도 본문 영역과 밀접한 관계가 있는 feature라고 할 수 있다.

따라서 새로운 시각적 feature로서 웹페이지의 중심인 화면 중심과 웹페이지 중심과의 HTML 노드의 거리를 제안한다. 이에 따라 중심 C와의 상대 거리 D를 정의한다. 각 HTML 노드의 중심 E에 대해서 중심 거리 D는 다음과 같다.

$$C = \{C_0, C_1\} \tag{1}$$

$$D = \{D_i \mid dist(E, C_i)\}$$

여기서 C_0는 사용자에게 보이는 화면(브라우저 창)의 중심이고 C_1는 렌더링 된 전체 웹페이지의 중심이다. *dist* 함수는 두 점 사이의 유클리드 거리를 구한다.

3-2 Google TabNet과 신경망 학습

Google TabNet[13]은 정형 데이터를 위한 딥러닝 아키텍처이다. 정형 데이터는 행과 열을 가진 표 형태로 나타낼 수 있는 데이터로 각 행은 개별 레코드를 의미하며 열은 데이터 타입이나 속성을 의미한다. 정형 데이터의 개별 레코드(행)는 동일한 수(열)의 데이터를 가지고 있다. 정형 데이터를 신경망에 학습시킬 때의 난점은 각 열이 동일한 데이터 비중을 갖고 있지 않으므로 적절한 열을 선택하는 것이다. 가령, 사람의 이름, 키, 몸무게 등의 개인 정보로 이루어진 정형 데이터에서 특정 질병의 위험도를 판단하는 모델을 적용하려 했을 경우, 이름이나 출생지보다는 나이, 성별, 키, 몸무게 등의 신체 정보를 가진 열이 질병과 관련이 있을 것이므로 더 비중 있게 학습되어야 한다. 따라서 정형 데이터는 각 셀(행, 열)의 데이터가 주변 데이터와 관계가 거의 없어 기존의 이미지나 텍스트 등 연속되는 값(픽셀 또는 단어 등)의 관계성이 중요한 비정형 데이터 학습에 높은 성능을 보였던 CNN과 RNN 등의 딥러닝 모델을 사용할 수 없었다. 또한 많은 경우 정형 데이터에는 희박한 데이터(sparse data)의 형태를 갖기 때문에 딥러닝 모델의 학습 정확도를 끌어올리기 힘들다.

정형 데이터의 신경망 학습 과정의 난점을 극복하기 위해 Google TabNet은 순차적 어텐션(Attention)을 사용하여 입력된 정형 데이터의 열에 순차적으로 학습과 피드백을 진행하며 feature selection을 수행한다. 이 과정을 통해 사람의 수동 개입 없이 각 결정 단계(decision step)에서 더 중요한 feature를 자동으로 선택한다. 따라서 학습 시, categorical 열은 별도로 지정하는 것을 제외하고는 전처리나 데이터 타입 선언이 필요하지 않다.

본 논문에서는 Google TabNet 아키텍처를 사용하여 두 가지 딥러닝 모델을 생성했다. 첫 번째 모델 TabNet_1 은 특정 웹페이지에 종속되거나 지역적, 언어적 특성이 없는 기본적인 텍스트, 구조적, 시각적 feature들을 학습한다. TabNet_1 이 학습한 feature 은 다음과 같다.

(a) 구조적 feature:
- DOM Tree에서 노드 순서 (preorder 탐색 기준)
- 태그 이름
- 자식 노드 수

(b) 텍스트 feature
- 텍스트 수
- 링크 텍스트인 (<a>에 포함된) 수

(c) 시각적 feature
- 노드 표시 여부 (visibility)
- 노드의 위치와 크기 (가로, 세로)
- 화면 크기 (가로, 세로)
- 웹페이지의 크기 (가로, 세로)

두 번째 모델 TabNet_2는 첫 번째 모델의 feature에 추가로 새로운 시각적 feature 인 상대 거리 feature D를 학습한다. 본 연구에서는 두 가지의 상대 거리 D 로 D_0과 D_1 값을 계산하여 학습 과정에 포함하였다.

모델 학습 시, 데이터로 DOM Tree의 HTML 노드를 하나의 레코드(행)으로 하고 3장 2-1절에서 설명한 feature를 각 열로 하는 정형 데이터를 구성하였다. categorical 열은 2종으로 태그 이름과 노드의 표시 여부(visibility)이다. 태그 이름은 Mozilla의 개발 문서[20]에서 제시하는 137개의 태그에 대응하고 그 외의 비표준 태그는 예외 태그 하나로 인코딩하여 138개로 변환하였다. 노드의 visibility는 jQuery의 :visible selector의 값으로 해당 노드가 시각적으로 표시되는 경우 (CSS 속성 등으로 인해 숨겨지지 않은 경우. 단, 다른 노드에 가려져서 보이지 않는지는 판단하지 않는다.) 는 1, 그렇지 않을 경우는 0으로 변환된다. 그 외의 feature는 실수형 숫자로 입력된다.

Ⅳ. 데이터셋 및 성능 실험

4-1 데이터셋 구축

주요 콘텐츠 추출 성능 비교를 위해 8개 언어로 된 웹페이지 데이터셋을 사용하였다(표 1)[21]. 이 데이터셋은 매년 각 지역의 대표 키워드를 발표하는 GoogleTrends 키워드의 검색 결과 중 상위 100개 웹페이지의 URL을 수집하였다(표 1의 URLs). 그 중에서 임의의 웹페이지를 수집하고(표 1의 Crawled) 실험 참여자가 직접 본문 영역을 태깅하였다(표 1의 Readable). GoogleTrends가 발표되지 않는 중국 지역에서는 중국 검색 포털인 Baidu에서 동일한 작업을 수행했다. 키워드 수집 대상은 2020년이나 글로벌 영역(영어)는 2017년이 추가로 수집되었다. 이 데이터셋에는 수집된 웹페이지에 대하여 수 명의 실험자가 제목, 본문, 기타 영역을 직접 태깅한 데이터를 포함하고 있다.

표 1. 실험 데이터셋
Table 1. Experimental dataset

Name	Region	URLs	Crawled	Readable
GoogleTrends-2017	Global	12720	390	285
GoogleTrends-2020	Global	10560	388	240
GoogleTrends-2020-KR	South Korea	296	43	21
GoogleTrends-2020-JP	Japan	450	47	24
GoogleTrends-2020-ID	Indonesia	900	50	31
GoogleTrends-2020-FR	France	1580	97	39
GoogleTrends-2020-RU	Russia	890	95	48
GoogleTrends-2020-SA	Saudi Arabia	260	97	43
Baidu-2020	China	1990	193	53

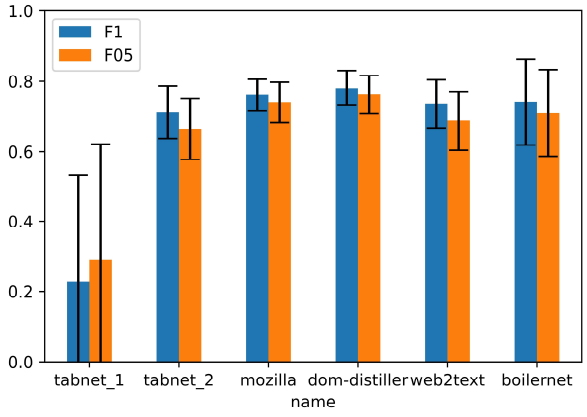

그림 1. LCS F-score 평균치 및 지역별 데이터셋 간 오차
Fig. 1. LCS F-score with error range by regional dataset

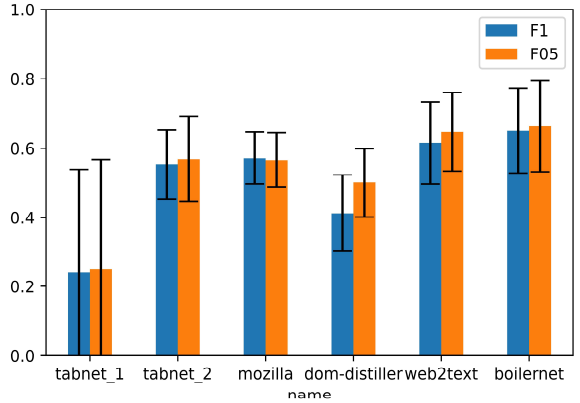

그림 2. Block F-score 평균치 및 지역별 데이터셋 간 오차
Fig. 2. Block F-score with error range by regional dataset

4-2 기술 간 성능 비교: F-scores

본 연구에서는 상기한 데이터셋의 글로벌 영역(GoogleTrends-2017, GoogleTrends-2020)의 웹페이지로 TabNet_1과 TabNet_2 모델을 학습하였다. Train:Valid:Test 비율은 80:10:10, 배치 크기는 1024 였다. 이 모델들을 유명 브라우저의 리더 모드 2종(Mozilla Readability.js, Google DOM Distiller)과 최신 기계학습 기반의 본문 추출 기법 2종(Web2Text, BoilerNet)과 비교하였다. 측정 기준은 본문 추출 기술에 자주 사용되는 척도인 longest common subsequce(이하 LCS)[22]와 matched text blocks(이하 Block)[4],[10]를 사용하였다.

각각의 본문 추출 기술들은 추출 결과로 웹페이지의 일부(HTML 노드) 또는 텍스트를 반환한다. HTML 노드를 반환하는 리더 모드 2종과 TabNet 2종에 대해서는 추출 결과와 데이터셋의 본문 영역을 정답 영역으로 취급하여 비교하는 것으로 Precision과 Recall을 측정한다. 즉, 가장 긴 문자열을 비교하는 LCS를 측정할 때는 추출 결과와 정답 영역의 텍스트를 비교한다. Block 측정의 경우 결과 노드와 정답 영역의 말단 노드(leaf node) 중 텍스트만 가진 블록을 조사하여 Precision과 Recall을 구한다. Web2Text와 BoilerNet은 결과로 텍스트의 배열을 반환하므로, LCS의 경우 결과 텍스트 배열을 하나의 긴 텍스트로 합하여 정답 영역의 텍스트와 비교하고, Block의 경우 결과 배열의 원소 텍스트와 동일한 텍스트를 가진 정답 영역의 말단 노드를 찾는 것으로 Precision과 Recall을 측정한다. 그 후, 각 추출 기술의 측정값으로 F-score 를 계산하여 비교하였다.

이 실험을 통해 특정 언어나 지역, 웹사이트에 Ad-hoc한 feature를 포함하지 않은 구조적, 텍스트 feature로 학습된 딥러닝 모델(TabNet_1)의 추출 성능을 살펴보고 동일한 feature 및 새로운 시각적 feature인 상대거리 D를 포함하여 학습된 모델(TabNet_2) 비교하고자 한다.

그림 1과 그림 2는 각각 LCS와 Block 측정 결과에 대한 F_1, $F_{0.5}$ 값의 평균과 지역 데이터셋별 오차를 표시한 결과이다. 표 2는 전체 실험 결과를 보여준다. F_1과 더불어 $F_{0.5}$ 값을 측정하는 이유는 $F_{0.5}$ 값을 통해 Recall 보다 Precision 에 더 비중을 둔 본문 추출 성능을 확인할 수 있기 때문이다. 가령, 본문 추출에 실패하였을 때, 웹페이지 전체를 결과로 반환할 경우 Recall이 1이 된다. 단, 브라우저 리더 모드의 경우 추출에 실패하더라도 본문의 일부를 유실하는 것보다 웹페이지 원본을 반환하는 것이 유리하기 때문에 $F_{0.5}$ 값이 낮더라도 반드시 성능이 부족하다고 할 수는 없다. 이러한 특징을 감안할 때 그림 1의 결과는 TabNet_1을 제외하면 LCS에서 F_1이 더 높은 것을 알 수 있다. 이는 대부분에 기술이 전체적으로 본문영역을 보다 과대하게 텍스트를 추출했음을 의미한다.

LCS 측정치에서 Readability.js(F_1 0.760)와 DOM Distiller(F_1 0.766)는 주요 브라우저의 읽기 모드로서 준수한 성능을 보이면서도 웹페이지에 언어로 인한 성능 변화가

표 2. 성능 측정 결과
Table 2. Experimental result

	Readability.js								DOM Distiller							
	LCS				Block				LCS				Block			
	Prec.	Recall	F1	F0.5	Prec.	Recall	F1	F0.5	Prec.	Recall	F1	F0.5	Prec.	Recall	F1	F0.5
2017	.761	.868	.743	.728	.571	.686	.580	.567	.760	.881	.749	.722	.705	.289	.339	.454
2020	.796	.905	.776	.765	.646	.766	.634	.625	.762	.886	.751	.729	.747	.369	.399	.494
KR	.783	.905	.784	.755	.514	.511	.511	.513	.831	.813	.753	.741	.848	.427	.492	.571
JP	.585	.988	.662	.609	.429	.625	.448	.435	.753	.921	.742	.712	.697	.353	.365	.437
CN	.757	.939	.787	.758	.525	.540	.519	.520	.862	.962	.876	.856	.869	.613	.653	.723
FR	.736	.911	.729	.696	.531	.673	.534	.515	.793	.879	.757	.741	.730	.239	.294	.400
SA	.803	.919	.775	.764	.630	.751	.641	.632	.769	.917	.748	.721	.760	.414	.420	.479
ID	.830	.914	.817	.814	.640	.636	.593	.603	.878	.909	.837	.830	.820	.263	.316	.425
RU	.843	.887	.773	.765	.721	.729	.687	.691	.886	.882	.812	.804	.849	.384	.433	.521
Avg.	.775	.897	.760	.744	.598	.698	.595	.586	.784	.891	.766	.743	.750	.351	.391	.482
	BoilerNet								Web2Text							
	LCS				Block				LCS				Block			
	Prec.	Recall	F1	F0.5	Prec.	Recall	F1	F0.5	Prec.	Recall	F1	F0.5	Prec.	Recall	F1	F0.5
2017	.680	.951	.751	.703	.708	.811	.696	.690	.564	.939	.651	.589	.553	.411	.382	.431
2020	.684	.952	.748	.705	.699	.817	.680	.675	.600	.959	.677	.619	.636	.810	.628	.620
KR	.846	.952	.887	.860	.898	.807	.834	.866	.672	1.000	.771	.705	.864	.716	.742	.785
JP	.459	.694	.477	.451	.447	.588	.406	.406	.663	.946	.714	.679	.701	.550	.485	.540
CN	.647	.721	.667	.654	.595	.579	.530	.547	.878	.936	.889	.881	.873	.722	.756	.803
FR	.690	.921	.755	.711	.735	.775	.703	.711	.644	.996	.739	.676	.696	.824	.679	.673
SA	.680	.881	.696	.662	.663	.736	.605	.599	.624	.931	.690	.645	.740	.688	.597	.642
ID	.843	.963	.853	.842	.849	.681	.684	.740	.694	.939	.746	.708	.709	.658	.610	.650
RU	.780	.940	.828	.796	.763	.806	.716	.730	.663	.917	.739	.676	.737	.703	.658	.681
Avg.	.689	.922	.745	.706	.702	.779	.671	.672	.621	.948	.695	.641	.647	.629	.547	.574
	TabNet_1 (Basic features)								TabNet_2 (Basic features + Visual features)							
	LCS				Block				LCS				Block			
	Prec.	Recall	F1	F0.5	Prec.	Recall	F1	F0.5	Prec.	Recall	F1	F0.5	Prec.	Recall	F1	F0.5
2017	.054	.082	.058	.054	.058	.069	.056	.055	.665	.917	.724	.683	.713	.702	.646	.667
2020	.038	.058	.042	.039	.045	.043	.037	.040	.684	.912	.726	.692	.700	.684	.622	.652
KR	.701	.944	.769	.722	.757	.654	.613	.644	.682	.899	.723	.687	.718	.562	.557	.612
JP	.109	.137	.086	.079	.037	.103	.048	.041	.688	.996	.772	.717	.591	.748	.597	.578
CN	.106	.125	.088	.080	.057	.057	.042	.047	.444	.958	.535	.471	.327	.645	.357	.331
FR	.107	.095	.073	.064	.027	.058	.029	.027	.592	.956	.704	.627	.460	.552	.426	.429
SA	.146	.169	.128	.123	.045	.105	.048	.042	.623	.933	.726	.653	.511	.641	.533	.515
ID	.671	.905	.695	.661	.697	.695	.590	.621	.648	.903	.687	.656	.728	.692	.611	.646
RU	.792	.924	.818	.799	.824	.703	.702	.750	.782	.904	.808	.790	.801	.620	.628	.697
Avg.	.177	.230	.183	.174	.175	.171	.150	.156	.668	.919	.724	.683	.681	.675	.608	.631

상대적으로 적었다. BoilerNet(F_1 0.745)과 Web2Text (F_1 0.695)는 두 종의 리더 모드와 유사한 성능을 보이지만 데이터셋의 지역에 따라 성능 차이가 두드러졌다. TabNet_1 (F_1 0.183)모델은 매우 낮은 성능을 보이며 데이터셋에 지역에 따라서도 성능 편차가 매우 컸다. 한편, TabNet_2(F_1 0.724)는 BoilerNet과 Web2Text와 비슷한 수준의 성능을 보였다.

Block 측정치에서는 Readability.js(F_1 0.595)와 DOM Distiller(F_1 0.391)는 BoilerNet(F_1 0.671)과 Web2Text (F_1 0.547)보다 낮은 성능을 보여주었다. 이는 읽기 모드 특성 상 웹페이지의 DOM Tree를 재구성하는 과정에서 기존 노드위치가 병합되거나 분리되는 경우가 발생하여 정답 노드와 매칭되는 노드를 찾지 못하는 경우 정확도가 떨어지게 되는 현상이 발생하는 영향이 있는 것으로 보인다. 그것을 감안하더라도 두 읽기 모드는 데이터셋의 지역에 따른 편차가 미세하게 적었다. TabNet_1(F_1 0.150)은 LCS와 마찬가지로 낮은 성능과 큰 지역에 의한 편차를 보였으며 TabNet_2(F_1 0.608)는 두 종의 읽기 모드와 두 종의 기계학습 모델에 사이의 성능을 보였다. 데이터셋의 지역 차이로 인한 편차는 비슷했다. 종합적으로 LCS와 Block 두 측정 방법에서 TabNet_1은 대부분의 데이터셋에서 추출에 실패할 뿐만 아니라 웹페이지의 지역, 언어 차이로 인한 큰 성능 편자를 보였으나 TabNet_2는 다른 비교 방법들과 비슷한 추출 성능 및 웹페이지 언어에 대한 성능편차를 보였다.

4-3 학습 모델 분석: Feature Importances

Feature Importance는 의사 결정 트리(Decision Tree)와 같은 기계학습 모델에서 특정 feature로 분기함으로써 얻는 성능적인 이득을 말한다. TabNet 또한 feature 선택 단계의 마스킹 과정에서 feature 분기에 의한 feature importance를 구할 수 있다.

그림 3과 그림 4는 각각 TabNet_1 모델과 TabNet_2 모델의 Feature Importance 그래프이다. 두 모델 공통적으로 기본 시각적 feature인 위치와 크기의 일부(top, bottom, width), 그리고 구조적 feature인 태그명(tagName)의 영향이 큰 것을 알 수 있다. 이는 본문 영역이 다른 영역과 구분되는 특유의 위치와 크기, 태그명을 가지고 있음을 보여준다. 또한 웹페이지의 본문 영역이 사용자의 가독성을 위해 상하 위치와 적절한 너비를 가지는 시각적 특성이 모델에 학습되었음을 의미한다. 마찬가지로 본문이나 텍스트를 표현하는 데에 사용되는 HTML 태그도 한정적이므로 태그명으로 인한 분기도 학습되었음을 보여준다. 그러나 상기한 특성들은 본문 영역에 포함되는 HTML 노드들의 대략적인 시각적 위치 또는 DOM Tree상의 구조적 위치를 파악할 수는 있으나 정확한 본문 영역을 결정하지는 못한다. 따라서 정확한 본문 영역을 결정할 수 있는 추가적인 feature가 필요하다.

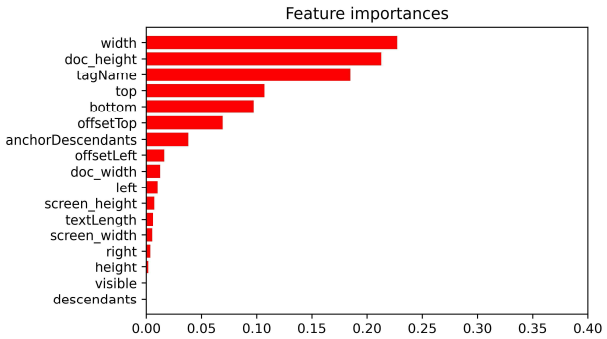

그림 3. TabNet_1 모델의 Feature Importances
Fig. 3. Feature importances of TabNet_1

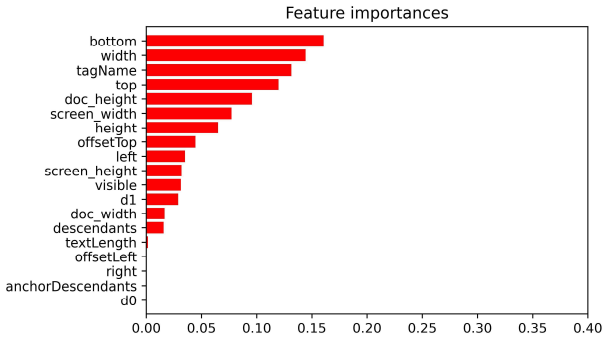

그림 4. TabNet_2 모델의 Feature Importances
Fig. 4. Feature importances of TabNet_2

본 논문의 실험에서 TabNet_1 모델과 TabNet_2 모델은 유의미한 성능 차이를 보였고 다른 Feature Importance가 비슷한 점을 고려하면 두 개의 새 feature가 본문 영역의 구체적 위치 확정에 영향을 끼쳤을 것으로 추측된다. 학습 단계에 따라 feature를 마스킹하여 학습하는 TabNet의 특성을 고려하면, 두 모델의 주요 분기는 비슷하나 이후 본문 영역을 결정하는 단계에서 주요하게 작용한 D_1 (d1) feature가 영향을 준 것으로 나타났다. 따라서 D_1과 유사한 역할을 하는 feature를 발굴하는 것으로 정확도를 높일 수 있음을 보여준다.

V. 결 론

본 논문에서는 웹 페이지의 주요 콘텐츠 추출 기술을 개선하기 위해 브라우저에서 렌더링되는 HTML 노드의 시각적 요소를 기반으로 하는 feature가 미치는 영향을 두 가지 TabNet 모델을 통해 살펴보았다. 성능 비교 결과 시각적 feature로서 웹페이지의 화면 중심과 문서 중심이 본문 추출에 큰 도움이 되는 feature임을 입증하였다.

본 연구의 모델 학습에 사용된 구조적, 텍스트 feature는 언어나 지역, 웹페이지 제작 시기의 영향이 적은 기초적인 속성만을 사용했으므로 특정 웹페이지의 템플릿 구조와 언어적 특성을 활용한다면 시각적 feature가 없어도 더 높은 성능을

가지는 모델을 훈련시킬 수 있을 것으로 예측된다. 그러나 웹 기술, 특히 웹페이지를 더 풍부하게 표현하기 위한 웹 프론트엔드 프레임워크 기술은 굉장히 빠르게 발전하는 영역이며, 지역별로도 인기가 있는 웹페이지 제작 방식(각 지역 지배적인 포털사이트나 서비스, Content Management System 등) 등이 다르기 때문에 구조적 feature들은 꾸준한 개발자의 관심과 유지보수가 필요하다. 마찬가지로, 모든 언어에 적용할 수 있는 텍스트 feature는 매우 한정적이기 때문에 각각 언어에 능통한 개발자들의 기여가 필요하다. 따라서 본 논문에서 제시하는 시각적 feature들은 인간의 본능적인 행동에 기반하는 것으로 특수한 목적을 가진 웹페이지(데이터 저장소 또는 광고, 마케팅 목적으로 본문이 모호하거나 의도적으로 숨겨지는 페이지)를 제외하면 언어나 지역 등의 조건에 연연하지 않고 높은 성능을 보일 수 있다. 다만, 이 feature는 웹페이지의 렌더링이 선행적으로 필요하므로 브라우저 또는 헤드리스 브라우저가 필요하기 때문에 다수의 웹페이지를 처리해야하는 경우에는 부적합할 수 있다. 그러나 브라우저의 리더 모드, 시각장애인을 위한 스크린 리더, 그 외에 브라우저 사용자에게 제공하는 서비스 구성에 효과적이다.

더불어, 본 연구의 Google TabNet을 활용한 본문 추출 시도는 웹페이지를 정형 데이터로 변환하고 이에 적절한 딥러닝 모델을 도입하여 웹페이지의 본문뿐만 아니라 특징적인 HTML 노드를 추출하는 데 사용할 수 있음을 시사하였다. 이를 통해 본 논문에서 제안하는 기법은 웹 페이지의 주요 콘텐츠 추출 기술에 대한 한계를 극복하고 더욱 정확한 추출을 가능하게 할 것으로 기대된다.

감사의 글

본 연구는 2016년도 및 2018년도 한국연구재단의 지원에 의하여 이루어진 연구로서, 관계부처에 감사드립니다. (2016R1A2B4016591, 2018R1A5A7059549).

참고문헌

[1] H. Han and T. Tokuda, A personal web information/knowledge retrieval system, in *Information Modelling and Knowledge Bases XIX*, Amsterdam: IOS Press, pp. 338-345, 2008.

[2] Y. Yesilada, “Web page segmentation: A review,” *EMINE Technical Report.* Middle East Tech. Univ. Northern Cyprus Campus. Deliverable 0 (D0), 1-39, March 2011.

[3] R. Fayzrakhmanov, E. Sallinger, B. Spencer, T. Furche, and G. Gottlob, “Browserless Web Data Extraction: Challenges and Opportunities,” in *Proceedings of the 2018 World Wide Web Conference*, Lyon, France, pp. 1095-1104, April 2018. https://doi.org/10.1145/3178876.3186008

[4] T. Vogels, O. Ganea, and C. Eickhoff, “Web2Text: Deep Structured Boilerplate Removal,” in *Proceedings of European Conference on Information Retrieval*, Grenoble, France, pp. 167-179, March 2018. https://doi.org/10.1007/978-3-319-76941-7_13

[5] A. Barbaresi, “Trafilatura: A web scraping library and command-line tool for text discovery and extraction,” in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, Virtual event, pp. 122-131, August 2021.

[6] R. Štrimaitis, P. Stefanovič, S. Ramanauskaitė, and A. Slotkienė, “Financial Context News Sentiment Analysis for the Lithuanian Language,” *Applied Sciences*, Vol. 11, No. 10, 4443, May 2021. https://doi.org/10.3390/app11104443

[7] H. Wan, W. Ji, G. Wu, X. Jia, X. Zhan, M. Yuan, and R. Wang, “A novel webpage layout aesthetic evaluation model for quantifying webpage layout design,” *Information Sciences*, Vol. 576, pp. 589-608, October 2021. https://doi.org/10.1016/j.ins.2021.06.071

[8] J. Martínez-González and C. Álvarez-Albelo, “Influence of Site Personalization and First Impression on Young Consumers’ Loyalty to Tourism Websites,” *Sustainability*, Vol. 13, No. 3, pp. 1-18, January 2021. https://doi.org/10.3390/su13031425

[9] G. Wagner, H. Schramm-Klein, and S. Steinmann, “Online retailing across e-channels and e-channel touchpoints: Empirical studies of consumer behavior in the multichannel e-commerce environment,” *Journal of Business Research*, Vol. 107, pp. 256-270, February 2020. https://doi.org/10.1016/j.jbusres.2018.10.048

[10] J. Leonhardt, A. Anand, and M. Khosla, “Boilerplate Removal using a Neural Sequence Labeling Model,” in *Proceedings of World Wide Web Conference 2020*, Taibei, Taiwan, pp. 226-229, April 2020. https://doi.org/10.1145/3366424.3383547

[11] M. Kim, Y. Kim, W. Song, and A. Khil, “Main Content Extraction from Web Documents Using Text Block Context,” in *Database and Expert Systems Applications*, pp. 81-93, August 2013. https://doi.org/10.1007/978-3-642-40173-2_10

[12] C. Kohlschütter, P. Fankhauser, and W. Nejdl, “Boilerplate detection using shallow text features,” in *Proceedings of the 3rd ACM International Conference on Web Search and Data Mining*, New York, NY, pp. 441-450, February 2010. https://doi.org/10.1145/1718487.1718542

[13] S. O. Arik and T. Pfister, “TabNet: Attentive Interpretable Tabular Learning,” in *Proceedings of the AAAI Conference on Artificial Inteligence*, Virtual event, pp. 6679-6687, February 2021. https://doi.org/10.1609/aaai.v35i8.16826

[14] A. Finn, N. Kushmerick, and B. Smyth, “Fact or fiction: Content classification for digital libraries,” *DELOS Workshops / Conferences*, 2001.

[15] GitHub - Mozilla/readability: A standalone version of the readability lib [Internet]. Available: https://github.com/mozilla/readability

[16] D. Cai, S. Yu, J. R. Wen, and W. Y. Ma, “VIPS: a Vision-based Page Segmentation Algorithm,” *Beijing Miciosoft Res. Asia,* MSR-TR-2003-79, November 2003.

[17] GitHub - chromium/dom-distiller: Distills the DOM [Internet]. Available: https://github.com/chromium/dom-distiller

[18] G. Lindgaard, G. Fernandes, C. Dudek, and J. Browñ, “Attention web designers: You have 50 milliseconds to make a good first impression!,” *Behaviour & Information Technology*, Vol. 25, No. 2, pp. 115-126, 2006. https://doi.org/10.1080/01449290500330448

[19] G. Jung, S. Han, H. Kim, K. Kim, and J. Cha, “Extracting the Main Content of Web Pages Using the First Impression Area,” *IEEE Access*, Vol. 10, pp. 129958-129969, December 2022. https://doi.org/10.1109/ACCESS.2022.3229080

[20] HTML elements reference [Internet]. Available: https://developer.mozilla.org/en-US/docs/Web/HTML/Element

[21] IEEE Dataport. Multilingual Datasets for Main Content Extraction from Web Pages [Internet]. Available: https://dx.doi.org/10.21227/rj0q-t583

[22] F. Sun, D. Song, and L. Liao, “DOM based content extraction via text density,” in *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Inforamation Retrieval,* pp. 245-254, December 2011. https://doi.org/10.1145/2009916.2009952

정근성(Geunseong Jung)

2014년 : 한양대학교 컴퓨터공학부 학사

2014년 ~ 현 재: 한양대학교 컴퓨터소프트웨어학부 석박통합과정

2022년 ~ 현 재: 임팩트에이아이 R&D 개발실 인공지능개발팀장

※관심분야 : 웹 콘텐츠 추출, 웹 데이터 마이닝, 웹 크롤링, 웹 자동화

차재혁(Jaehyuk Cha)

1987년 : 서울대학교 계산통계학과 학사

1991년 : 서울대학교 컴퓨터공학과 석사

1997년 : 서울대학교 컴퓨터공학과 박사

1998년 ~ 현 재: 한양대학교 컴퓨터소프트웨어학부 교수

※관심분야 : 데이터베이스, 플래시 스토리지, 멀티미디어 콘텐츠 적응