Information retrieval

Final Report

-여기 우리 앱 이름이 들어갔으면 좋겠음(html, no-td model, web-based, visualizer ~~)-

12.06/2015

Team 6

Computer science 2013210085 Sangho Suh

Industrial engineering 2010170835 Moonshik Kang

Computer science 2010210012 Hoonhee Cho

Contents

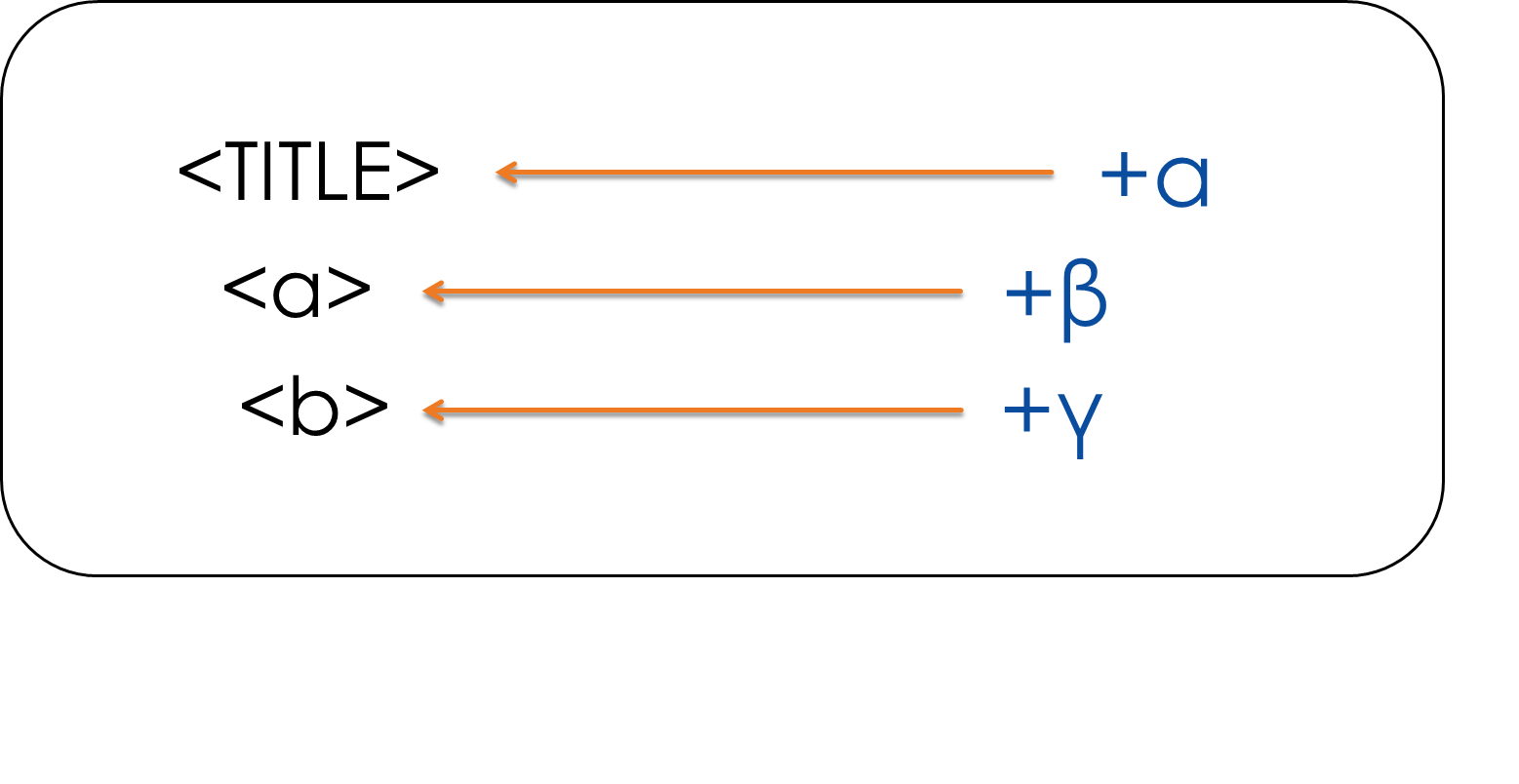
* Introduction
* Proposal Recap
* Implementation
* Evaluation
* Future Work
* References
* **Introduction (문제 도출, 전체 과정 개괄 설명, abstract)**

In order to make data visualization not only accessible and relatable but also useful to general users, we plan to make a tool that can be entertaining and useful for general users’ daily tasks. Thus, we propose *a data visualization tool[1]* whose primary function is providing keywords from current page’s(현제 페이지의 정보를 검색한다는 것을 강조) internet wikis, news, or articles. It will help simplify the process of rummaging through information presented in the form of text in the internet. With this tool, users can instantly get a summary of the corresponding internet news or article by identifying their significant keywords.

정보검색에서는 일반적으로 가장 많이 사용하는 데이터는 term-documents matrix이다. A document-term matrix or term-document matrix is a mathematical matrix that describes the frequency of terms that occur in a collection of documents. [2] 하지만 우리의 프로젝트에서는 현재 페이지 내의 정보를 통해 (그러니까 제한된 상황과 데이터안에서 under sparse text constraint) 토픽이나 키워드를 끄집어내야 하는 제약사항이 존재한다. 우리는 이것을 극복하여 중요한 키워드를 사용자에게 간편하고 빠르게 제공하는 **크롬 익스텐션 어플리케이션을** 제작하고자 한다. 또한 그것에 대한 비교 실험과 다양한 평가를 통해 정보검색의 질을 평가해보고자 한다.

* **Proposal Recap (전반적인 프로젝트 내용)**

그렇다면 우리는 어떤 정보를 이용할 수 있을까? 가장 기본적인 것은 워드-count 자료 구조이다. 페이지 안에서 등장하는 단어의 횟수가 간단히 그 중요도를 나타내는 척도로써 사용될 수 있을 것이다. [3]

 Fig 2-1. Weighting skeleton based on Web page tag

두번째는 태그의 이용이다. 웹 페이지는 html의 태그 언어로 이루어져있다. 이 태그는 어느정도 글과 관련된 정보를 가지고 있다. 예를 들면 <TITLE>태그는 일반적으로 그 페이지의 주제를 담고 있으며 그 태그 안에 있는 워드는 모든 페이지를 아우르는 키워드를 가지고 있음이 자명하다. 또한 <b>태그는 단어를 굵게 처리하는 태그로써 작성자가 중요하다고 생각하는 단어를 bold처리 했을 것이다. 이러한 예측들을 이용해 모든 워드에 동일한 count값을 주는 것이 아닌 각각의 가중치를 다르게 하여 계산할 수 있으며 이 가중치를 여러 케이스별로 실험하여 최적의 weight값을 찾을 수 있을 것이다. 이 아이디어에 대한 간단한 스케치는 Fig 2-1에서 볼 수 있다. 알파 베타 감마 값을 다양하게 실험해보고 사용자 평가를 거쳐 제일 키워드를 잘 뽑아낼 수 있는 어플리케이션을 만들 수 있다.

세번째로 제목으로부터의 거리이다. 위 태그 분석을 통해 글의 타이틀이 어디 있는지 알아내었다. 일반적으로 타이틀과 거리가 가까운 단어들은 글의 주제를 담고 있을 확률이 높으며 키워드로인정하기 충분하다. 이 정보를 발전시키면 타이틀에 가까운 단어(abstract에 있는 단어들)들은 가중치를 더 주고 멀리 있는 단어는 비교적 적은 가중치를 줄 수 있을 것이다.

이외에도 위 과정을 통해 이루어진 랭킹을 다시 재사용하여 상위 랭킹 3개 단어를 다른 검색사이트를 이용하여 검색 후 다시 그 페이지에 있는 것을 워드 카운트에 추가 하여 ‘피드백’과정을 만들어 낼 수 있다. 또한 거리 정보이외에도 **Ontology based Similarity를 이용해 주제어와 비슷한 단어들에 가중치를 더 주는 방식도 있겠다.**

**위의 방식들을 이용해 강화된 백오브워드 모델을 아주 펜시한 모습으로 유져에게 보여져야 할 것이다. 우리의 목표는 유저가 빠르고 간단하게 페이지전체의 내용을 인식하게 하는 것이기 때문이다.**



Fig 2-2. chrome based visualizer example (디자인후 교체 요망)

우리가 생각한 모습은 위(Fig 2-2)와 같다. 글의 순서는 유지하면서 각 단어마다 주어진 가중치에 따라 텍스트의 크기와 색이 다르게 보여진다. 중요한 단어는 크고 진한 색, 덜 중요한 단어는 작고 흐린 색으로 이루어져있다. 유저는 이를 통해 직관적으로 페이지 전체의 중요 키워드를 빠르게 찾을 수 있을 것이다.

* **Implementation (구체적인 구현 내용)**

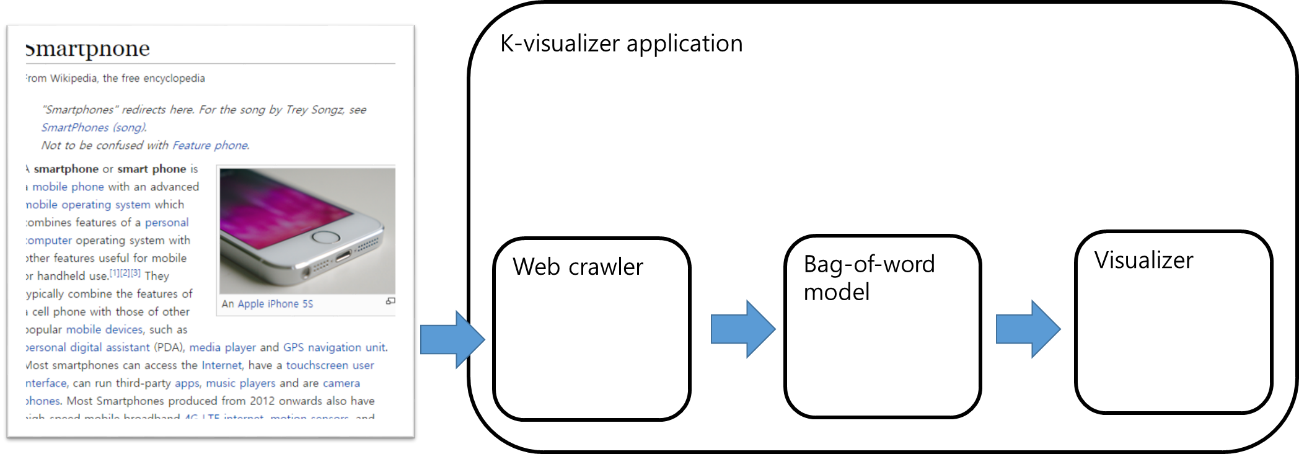
****

Fig 3-1. Overall architecture

Our k-visualizer application is a chrome extension app that can be easily accessed while general users are surfing the webs. The main mechanism in the app is separated by three parts, Web crawler, Bag-of-word generator, and Visualizer. 그리고 각각이 하는 일을 순서 대로 정리해보면 다음과 같다.

1) Clean your text

(Remove punctuations and stop words)

2) Tokenize the text

3) Find the TF (term frequency) for each unique token

4) Sort each token in descending count order

They are fundamentally implemented by JavaScript, PHP, and html. The current situation of each method will be explained below.

* 1. **Web page crawler**

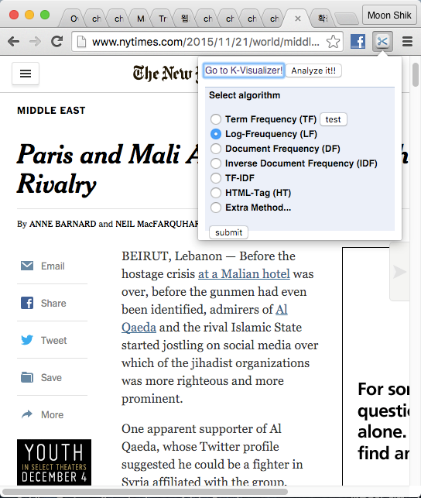
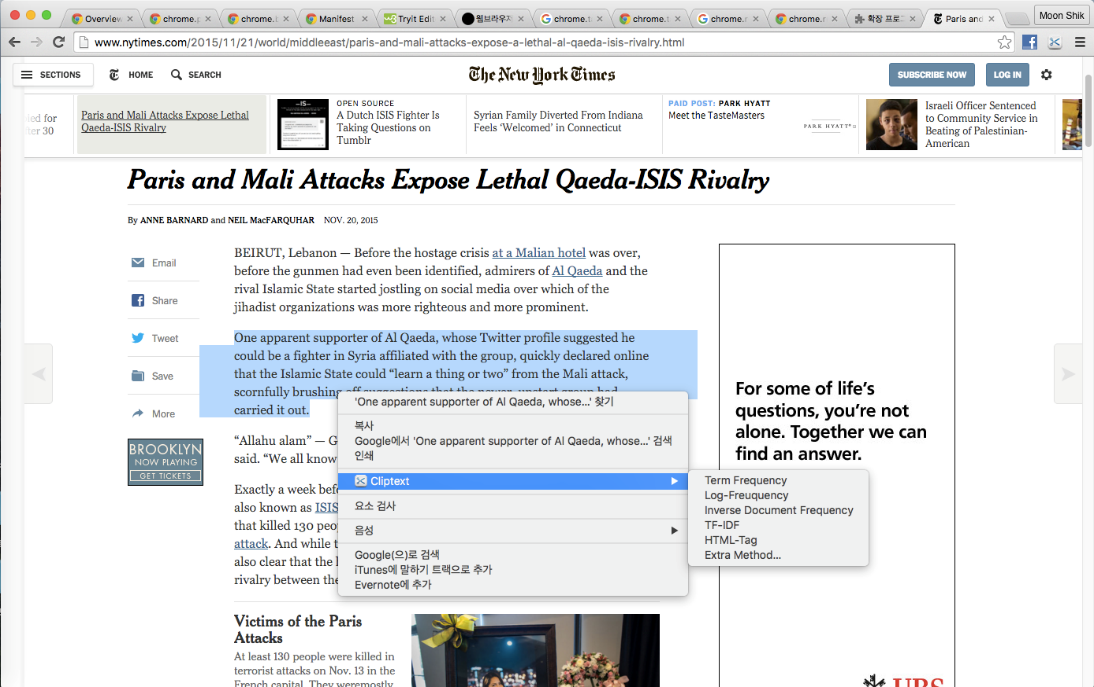
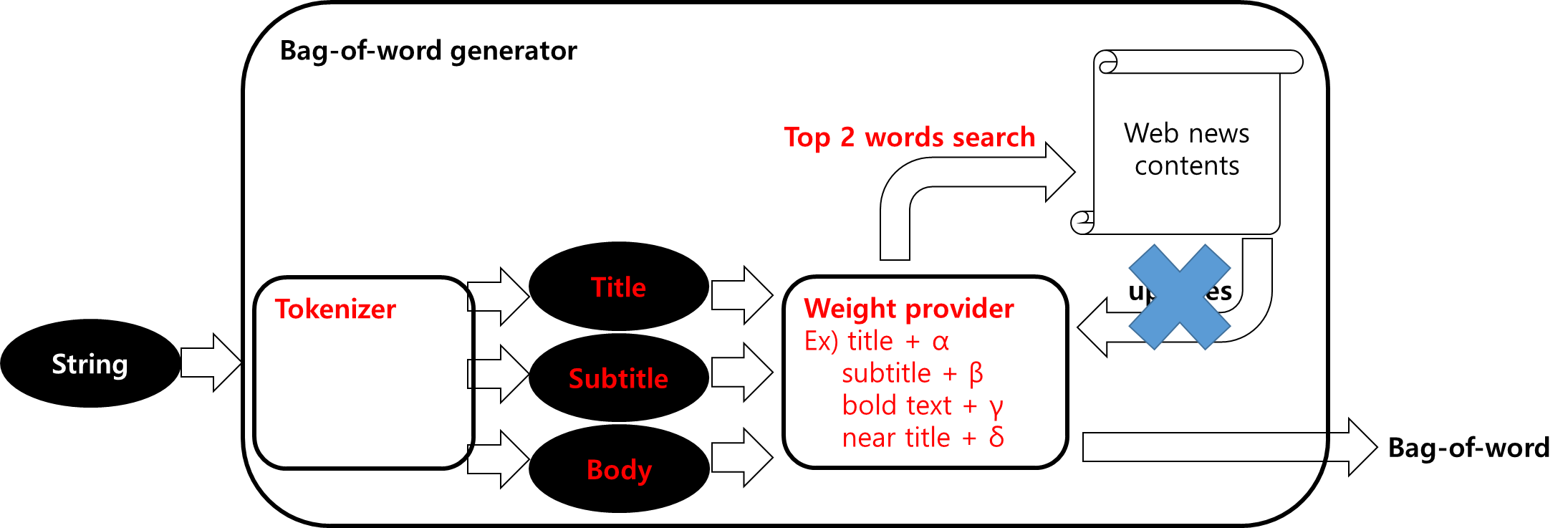
 

Fig 3-2. Process of crawling문식아 이거 지금 버전으로 사진 바꿔줘

With the plugin comes two interfaces. The first one is in popup page (left), and the other one is in right click menu (right). We tested these two methods and chose the first, because we received feedbacks from test users that the first one is more intuitive and better in terms of user experience (UX).

Also, with each method in plugin, we can apply many kinds of algorithm to analyze the web page. Thus, we are able to compare each method and visualize them in this plugin.

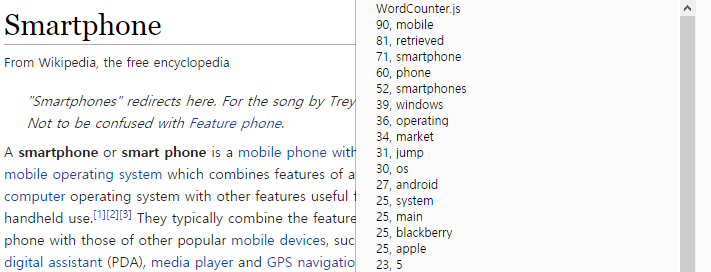
* 1. **Bag-of-word generator**

*Fig. 3.3. Bag-of-word generator abstract diagram*

*Bag-of-word generator* makes a word-frequency matrix that has tokenized words of the string (crawled string) and its word count. In addition to this, it can give each token customized weight. For example, as you can see in *Fig. 2.1,* the words that consist of title, subtitle, and bold text get higher weight (50, 30, 20) than normal text (weight is 1 for normal text).

***Tokenizer****:* input crawled string is a single string, so has to be split into word chunks. Therefore, the output is array format that is “wordcount[word]”.

***Weight provider*:** tokenized chunks array is an input that is simply separated as Title, Subtitle, and Body. After that, each word will be counted one by one, and be given the weight according to their categories



*Fig. 3.4. output of bag-of word Model*

* 1. **Visualizer**

There are two websites, one hosted by Parse and the other by Heroku. Server hosted by Parse supports MySQL and PHP, and Heroku server uses Node.js. It was unclear which backend we would end up using, so we have two website hostings with different backend framework. Depending on which backend framework is chosen after experimenting with PHP and Node.js, we can simply upload HTML, CSS, Javascript files.

Also, after receiving text data sent by chrome plugin (see ‘Web Page Crawler’ above), we have a visualizer (see below), which visualizes the crawled texts in wordle format, which is implemented in Javascript.

* **References (혹시 문식이꺼에도 레퍼있음 같이 합치면 됨)**

[1] Creative Bloq. "The 37 best tools for data visualization". *http://www.creativebloq.com/design-tools/data-visualization-712402*. Nov 11, 2014

[2] Wikipedia. “Document-term matrix”. [*https://en.wikipedia.org/wiki/Document-term\_matrix*](https://en.wikipedia.org/wiki/Document-term_matrix)*.* 14 November 2015

[3] Olena Medelyan, Ian H. Witten and David Milne. 2008. “Topic Indexing with Wikipedia”. *University of Waikato.* pp. 22