[2014년도 소프트웨어개발전공]

프로젝트
최종 보고서

프로젝트 제목	국 문 : 미어캣			
	영 문 : meerkat			
개 발 기 간	2014 년 4 월 1 일부터 2014 년 09 월 30 일까지 (6 개월)			
팀 구성원	구 분	이 름	E-mail	H.P
	팀장	이상협	mia22rmrjs9 @gmail.com	010-2637 -9084
	팀원	동승일		
		이상민		
		정해덕		
		김학성		

2014 년 11 월 12 일

신흥대학 컴퓨터정보계열

소프트웨어개발전공

【 목 차 】

문서 추가/수정 내역

원안작성자	정해덕, 이상협, 김학성
수정작성자	정해덕, 이상협, 김학성

수정날짜	수정자	추가/수정 항목	내 용
2014-10-29	정해덕	추가	2.1~2.2, 2.6절 작성
2014-10-29	이상협	추가	2.4, 2.5절 작성
2014-10-29	김학성	추가	1장 작성
2014-10-30	정해덕	수정	1장 오탈자 및 문맥 교정
2014-10-30	이상협	수정	2.1~2.3, 2.6절 오탈자 및 문맥 교정
2014-10-30	김학성	수정	2.4, 2.5절 오탈자 및 문맥 교정
2014-10-31	이상협	추가	4장 추가
2014-11-6	정해덕	수정	3.1절 수정
2014-11-6	이상협	추가	2.3절 내용 추가
2014-11-6	김학성	추가	2.7절 내용 추가
2014-11-10	정해덕	수정	3.1절 수정
2014-11-12	이상협	수정	목차 수정

1. 개요

1.1 프로젝트 개요

최근 동향에 따르면 독도를 비롯해 일본의 위안부 문제를 통해 시민들의 역사에 대한 인식이 증가하였다. 그러나 이러한 역사에 대한 지식을 알고자 하는 경우에 검색엔진을 통해서 정보를 얻어야 하지만 이러한 정보들은 단편적인 정보만이 제공이 된다. 사용자는 검색 후 본문 내용에서 검색 키워드에 관한 또 다른 지식(키워드)을 알지 못하는 경우 해당 키워드를 통해 재검색해야 하는 불편함을 가지고 있다. 이러한 불편함을 덜고자 본 프로젝트에서는 미어캣(meerkat)이라는 검색엔진을 제공하며, 제공되는 정보의 분야를 국사 분야로 정의하여 국사에 해당하는 정보를 검색을 했을 때 그것에 관한 지식 정보를 정확하게 보여준다. 또 해당 키워드에 대한 또 다른 지식(키워드)을 지식 간의 관계로 형성하여, 지식지도 형태로 보여주는 웹 기반의 서비스이다.

1.2 프로젝트 목표

사용자가 단편적인 정보를 얻고자 할 때 현존하는 검색엔진에서 충분한 정보를 얻을 수 있다. 그러나 복잡한 개념에 대해서 학습하고자 할 때에 관련된 지식(키워드)에 대해 추가적인 검색을 필요로 한다. 본 프로젝트의 목적은 이러한 상황에서 사용자가 좀 더 효율적으로 학습 정보를 습득할 수 있도록 지식 서비스를 제공하고자 한다. 문서와 문서 사이의 연관관계를 나타내고 지식지도 형태소로 나타내어 이를 통한 사용자의 효율적인 학습을 도모하는 것을 목적으로 한다.

1.3 추진 배경 및 필요성

- **국사에 특화된 검색엔진을 제공하기 위한 프로젝트**

사용자들은 원하는 정보를 검색할 때 검색엔진을 통하여 정보를 정확하고 좀 더 쉽게 얻을 수 있기를 바라고 있다. 본 프로젝트에서는 국사 분야의 특화된 정보 검색 기능과 검색한 키워드가 없는 경우, 그리고 오타가 발생하는 경우 검색 키워드의 유사 키워드를 추천해주는 기능을 제공한다.

- **효율적인 학습을 위한 지식지도 제공**

조선건국역사에 대한 개념을 좀 더 쉽게 이해하려면 고려 말의 역사를 알아야 하듯이, 한 가지 지식을 학습할 때 그 지식에 관련된 선행 지식 또는 후행 지식을 안다면 학습을 하는데 효율적으로 정보를 습득할 수 있다. 본 프로젝트는 국사에 관련된 여러 인물 및 사건과 역사에 대한 내용들을 각각의 키워드로 만들어, 관련 있는 내용의 키워드를 서로 연결 지어 지식지도를 생성한다.

2. 개발내용 및 결과물

2.1 시스템 기능

2.1.1 지식검색

- 문서내의 키워드를 통한 검색

기존 웹상에서 제공되는 검색서비스와 같이 사용자가 검색할 키워드를 입력시 검색조건에 부합하는 서비스를(검색 기능을) 제공한다. 또한 사용자가 키워드를 입력하였을 때 해당 문서가 존재하지 않거나 잘못 입력하였을 때 유사 키워드를 추천해준다.

2.1.2 지식지도생성

- 지식간의 관계를 통해 지도를 생성

관계가 맺어진 문서를 Springy를 통해 지식지도로 생성하여 보여준다. Springy란 그래프 시각화 중 한 가지 방법으로써 Java Script를 이용하여 만들어진 API이다. 사용자가 단순히 검색을 통하여 얻고자 하는 정보뿐만 아니라 지식지도를 보여주어 지식 간의 관계를 한눈에 파악할 수 있으며 연결된 키워드를 누르면 또 다른 지식 간의 관계를 나타내주어 연관된 학습을 할 수 있도록 도와준다.

2.1.3 웹문서 수집

- 소스코드 형태의 웹문서로부터 본문 추출

웹을 통해 제공되는 국사 분야 페이지의 경우 필요한 정보뿐 아니라 많은 불필요한

정보를 담고 있다. 이러한 불필요한 정보는 시스템의 성능 저하와 비효율성을 가지고 온다. 또한 지식지도 지도 작성 시 불필요한 요소들은 배제하여 최적화된 서비스를 제공하고자 한다. 그리하여 본 과정에서는 필요 부분인 본문의 정보만을 추출한다. 정보 추출시 Goose-Extractor를 이용하여 추출한다.

- 추출된 본문의 형태소 분석

자연 언어 처리에서 말하는 형태소 분석이란 어떤 대상 어절의 모든 가능한 분석 결과를 출력하는 것을 의미한다. 형태소 분석을 통하여 미 등록어, 오탈자, 띄어쓰기 오류 등을 분석한다. 본 프로젝트에서는 KLT-2010 형태소 분석기를 이용하여 분석하였다.

2.1.2. 문서분류 및 유사도 계산

- 문서의 이진 분류

본 프로젝트에서는 국사 분야의 문서를 대상으로 지식지도를 작성하기 때문에 해당 문서가 국사 분야인지 아닌지를 판단하기 위해, 이진 분류하여 문서를 필터링한다. Naive Bayesian Classifier를 이용하여 추출한 문서가 어떤 분야에 속할 가능성이 높은가를 예측하는 확률적인 방법이다.

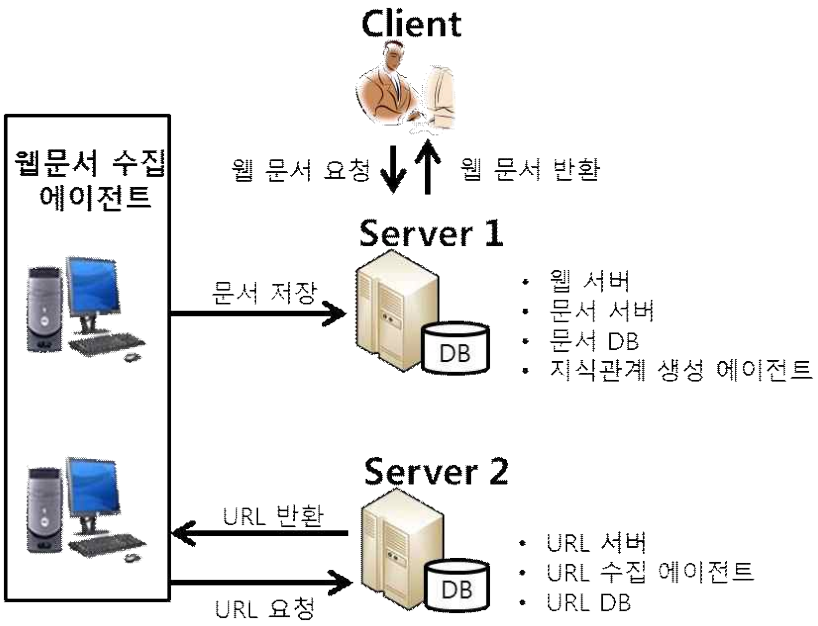
- 추출된 주제어를 통한 문서의 그룹화

문서의 주제어 추출 시 주성분 분석(PCA)을 이용하여 추출한다. 주성분 분석을 이용하여 문서의 키워드를 축소하고 정보의 손실을 최소하면 소수의 몇 개 단어로 문서의 내용을 표현할 수 있다. 이러한 방법을 이용해 주제어를 추출한다. 그리고 주제어 분석을 통해 많은 빈도수를 차지하는 단어가 그 문서의 주 키워드가 되며 주성분 분석을 통해 얻어지는 키워드를 가지고 관련된 문서와 그룹화를 한다.

- 그룹화 된 문서간의 유사도 측정을 통한 지식관계 생성

지식 관계 생성 시에 Cosine Similarity를 이용하여 유사도 점수를 계산한다. 관계를 생성할 두 지식 간에 존재하는 웹문서들을 이용해 벡터공간을 생성하고 벡터 공간 모델로부터 계산한 유사도 점수에 따라 지식 간의 관계를 생성한다.

2.2 시스템 구조



노드명	역할	구성요소 명	역할
웹문서 수집 에이전트	수집한 URL을 문서 형식으로 Server1에 저장		
Client	웹 브라우저를 통해 meerkat을 사용하는 이용자.		
Server 1	문서와 관련된 CRUD를 수행하고 지식관계를 생성하는 프로그램들이 작동되는 물리적 노드.	웹 서버	작성된 지식지도를 웹문서와 함께 클라이언트에게 제공.
		문서 서버	다수의 에이전트로 부터의 문서 DB 접근을 조율.
		문서 DB	수집된 문서를 저장.
		지식관계 생성 에이전트	수집된 웹문서간의 유사도를 측정하여 지식간의 관계를 생성.
Server 2	URL과 관련된 CRUD를 수행하는 구성요소들이 실행되는 물리적 노드.	URL 서버	다수의 에이전트로 부터의 URL DB 접근을 조율.
		URL DB	수집된 URL을 저장.
		URL 수집 에이전트	웹문서 수집 에이전트에게 제공할 URL을 수집하는 역할.

2.3 개발/운영 환경

2.3.1 소프트웨어

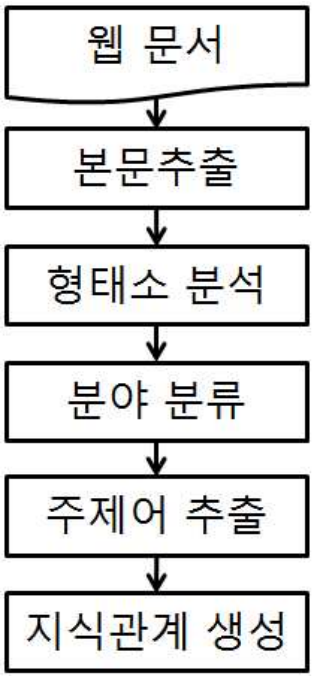
소프트웨어 명	버전	용도
Ubuntu Linux	14.04 lts	웹 서버와 웹문서 수집 에이전트의 운영체제
Crunchbang Linux	11	웹문서 수집 에이전트의 운영체제
Debian Linux	7.0	Url Server의 운영체제
Nginx	1.6	웹 서버
PostgreSQL	9.3	데이터베이스 서버
Subversion	1.8.8	소스코드관리
GNU Emacs	24.3.1	텍스트 편집
Eclipse	4.4.0	통합개발환경

2.3.2 외부 모듈

모듈명	버전	구현언어	용도
Django	1.6.1	Python	웹 서버 구현 프레임워크
Springy	2.6.1	JavaScript	지식지도를 시각적으로 구현
BootStrap	3.0	JavaScript	반응형 웹페이지 구현
PyTagCloud	0.3.5	Python	메인 페이지의 Tag Cloud 생성
KLT2010-TestVersion	2.0	C	한글 형태소 분석
goose-extractor	1.0.20	Python	웹 페이지의 본문 추출
sqlalchemy	0.9.7	Python	ORM 프레임워크
mdp	3.3	Python	주성분 분석
numpy	1.9.0	Python	행렬연산

2.4 연구/개발 내용

2.4.1 지식지도 작성절차



지식지도를 작성하기 위해서는 크게 5단계의 과정을 거치게 된다. 첫 번째 과정은 Crawler가 소스코드 형태의 웹 문서에서 본문을 추출하는 것이다. HTML 또는 XML로 작성된 웹 문서의 본문 추출에는 Python 언어로 구현된 오픈소스 모듈인 Goose-Extractor가 사용되었다. 다음으로는 추출된 본문에 대해서 형태소 분석을 실시한다. 한국어 형태소 분석기로는 국민대학교에서 개발된 형태소 분석기인 KLT-2010을 사용하였다. 형태소 분석 이후에는 추출된 형태소 분석결과를 가지고 해당 문서가 국사분야의 문서인지 아닌지 여부를 이진 분류한다. 문서의 이진 분류에는 Naive Baysian Classifier를 사용하여 국사분야가 아닌 문서를 필터링 하였다. 이후에는 국사분야로 분류된 문서에 대해서 주제어를 추출하여 해당문서를 추출된 주제어에 속하는 문서로서 그룹화하여 DB에 저장한다. 주제어 추출에는 주성분분석을 이용하여 웹문서의 주제어를 추출하였다. 지식지도 작성을 위한 마지막 단계는 주제어로 그룹화된 문서들 간의 관계를 생성하는 것이다. 지식관계의 생성에는 Cosine Similarity를 이용한다.

2.4.2 Crawling

본 프로젝트에서는 웹상에 존재하는 비정형의 웹문서들을 대상으로 하여 동적인 지식지도를 작성하는 것을 목표로 한다. 때문에 웹상에 존재하는 문서를 수집하기 위한 Crawler가 필요한데, 이를 위한 Crawler와 URL수집기는 Python언어로 구현하였다.

Crawling은 Seed를 입력하는 것에서부터 시작된다. Seed란 Crawling을 수행할 때 시작점이 되는 URL을 말하는 것으로서, 본 프로젝트에서는 수작업으로 직접 수집한 3개의 URL을 Seed로서 사용하였다.

대상 사이트	Seed URL
위키피디아	http://ko.wikipedia.org/wiki/%EB%B6%84%EB%A5%98:%ED%95%9C%EA%B5%AD%EC%9D%98_%EC%97%AD%EC%82%AC
네이버 지식백과	http://terms.naver.com/list.nhn?cid=41703&categoryId=41703
엔하위키미러	https://mirror.enha.kr/wiki/%ED%95%9C%EA%B5%AD%EC%82%AC%20%EA%B4%80%EB%A0%A8%20%EC%A0%95%EB%B3%B4

Seed가 입력되면 URL수집기는 해당 URL로 접근하여 웹페이지를 Parsing해서 모든 링크를 수집한다. 이후에는 수집된 링크들에 대해서 너비우선탐색방식으로 다음 URL에 차례로 접근하여 최종적으로는 해당 사이트 내의 모든 URL을 수집하게 된다.

Crawling의 마지막 단계는 Crawler가 수집된 후보URL들에 접근하여 필요한 정보를 추출하는 것이다.

Crawler는 URL수집기가 수집한 URL들을 가지고 URL접근, 본문 추출, 형태소 분석, 이진분류, 문서수집, 주제어추출의 과정을 순서대로 거쳐서 최종적으로 문서를 수집하게 된다.

2.4.3 웹 페이지의 본문 추출

```
>>> from goose import Goose
>>> from goose.text import StopWordsKorean
>>> url='http://news.donga.com/3/all/20131023/58406128/1'
>>> g = Goose({'stopwords_class':StopWordsKorean})
>>> article = g.extract(url=url)
>>> print article.cleaned_text[:150]
경기도 용인에 자리 잡은 민간 시험인증 전문기업 ㈜디지털이엠씨(www.digitalemc.com).
14년째 세계 각국의 통신·안전·전파 규격 시험과 인증 한 우물만 파고 있는 이 회사 박채규 대표가 만나기로 한 주인공이다.
그는 전기전자·무선통신·자동차 전장품 분야에
```

Python언어로 구현된 Goose-Extractor는 영어뿐만 아니라 한글, 중국어, 아랍어 등으로 작성된 비정형의 웹 페이지에서 본문, 메인 이미지 등의 주요 컨텐츠를 몇 줄의 코드만으로 간단하게 추출할 수 있다.

2.4.4 한글 형태소 분석

형태소 분석을 위해 사용된 KLT-2010 형태소 분석기는 국민대학교에서 C언어를 통해 개발된 한국어전용 형태소 분석기이다. Python언어는 여러 개의 거대한

소프트웨어 컴포넌트를 엮어서 사용하기 편리한 Glue언어라고도 불리는 만큼, Python언어로 작성된 프로그램에서 C 또는 Java와 같이 타 언어로 작성된 모듈을 사용하기 위해서는 SWIG나 Python C API등과 같은 다양한 방법이 존재한다.

```
import commands

analyzer_call_command = 'cd KLT2010-TestVersion/EXE && ./indexT ../../temporary_data/test.txt'
analyze_result = commands.getoutput(analyzer_call_command)
```

본 프로젝트에서는 단순히 형태소 분석기를 작동시키고 결과 값을 읽어오기만 하는 것이 목적이므로, C언어로 작성된 형태소 분석기를 Python 프로그램 내에서 리눅스 명령어로 직접 실행시키는 방법을 통해 구현하였다. KLT-2010의 입력 값으로 텍스트를 전달할 때에는 해당 텍스트를 '.txt''파일로 변환하여 전달하였다.

```
이것은 형태소 분석기의 사용 예제이다.
```

위와 같은 문장으로 구성된 문서를 형태소 분석기에 입력하면 아래와 같은 분석결과를 얻을 수 있다.

No	Freq	Score	Term
1	1	1000	형태소
2	1	836	사용
3	1	785	분석기
4	1	523	이것
5	1	135	예제이다
6	1	65	이다
7	1	65	예제

형태소 분석결과에서 No는 키워드의 중요도 순서를 의미한다. Freq는 키워드의 빈도수로서 해당 문서에서 키워드의 등장 횟수를 의미한다. Score는 키워드의 중요도 점수를 의미하는 것으로서 No가 가장 높은 키워드가 가장 높은 점수를 갖는다. Term은 추

출된 형태소를 의미한다. 본 프로젝트에서는 문서의 이진분류와 유사도 측정을 위해서 형태소 분석결과에서 Freq와 Term만을 사용한다.

2.4.5 문서의 이진분류

특정분야의 지식지도를 작성하기 위해서는 문서를 수집하는 과정에서 해당분야에 속하지 않는 문서들을 필터링할 필요가 있다. 본 프로젝트에서는 국사 분야만을 대상으로 지식지도를 작성하므로, 문서에 대해서 국사분야인지 아닌지 여부를 이진분류하여 문서를 필터링한다. Naive Bayesian Classifier는 지도학습(Supervised Learning)된 데이터를 기반으로 Bayes의 정리를 이용 하여 새로운 문서가 등장하였을 때, 어떠한 분야에 속할 가능성이 높은가를 예측하는 확률적인 방법이다.

나이브 베이지안 분류기를 구현하기 위한 과정은 크게 세 단계로서, 학습데이터 수집, 알고리즘 학습, 문서 분류로 나눌 수 있다.

- 학습데이터 수집

문서의 이진분류를 위해서 분류기를 학습시키기 위해서는 국사 분야의 문서들과 그렇지 않은 분야의 문서들이 각각 필요하다. 본 프로젝트에서는 학습데이터를 수집하기 위해서 위키피디아로 부터 4000개의 국사분야 문서와 국사 분야가 아닌 8개의 분야에서 4000개의 문서를 수집하여, 국사분야와 그렇지 않은 분야에서 총 8000개의 문서를 수집하여 분류기를 학습하였다.

- 알고리즘 학습

나이브 베이지안 분류기를 학습한다는 것은 학습문서들을 형태소 분석하여 각각의 키워드가 등장한 문서의 개수를 데이터로서 저장하는 것을 의미한다.

형태소	등장한 문서 수
왕침	2
문종	120
태후	126

저장된 데이터의 형식은 '.txt'파일에 위의 표와 같은 내용구성의 데이터를 '/'를 구분자로 하여 저장하였다. 예를 들어 형태소 '왕침'의 경우 '왕침/2'와 같은 형식으로 데이터를 저장하였다. 4000개의 국사분야의 문서들에서 형태소 분석된 키워드는 총 94012개 이며, 4000개의 국사분야가 아닌 분야의 문서들에서 형태소 분석된 키워드는 총 96710개 이다.

- 문서 분류

문서 분류를 위하여 다음과 같은 나이브 베이지안 공식을 적용하였다.

$$P(c_j|d) = \frac{P(C_j)P(d|C_j)}{P(d)}$$

여기에서 d는 임의의 웹문서를 의미하고, C_j는 문서의 분야를 의미하는 것으로, 국사분야와 그렇지 않은 분야의 2가지가 존재한다. P(d)는 모든 분야에 대해서 같은 값을 가지므로 학률을 계산하는데 있어 고려하지 않아도 된다. 따라서 P(Cj)와 $P(d|C_j)$만 알면 웹문서가 국사분야에 속할 확률과 그렇지 않은 확률을 구할 수 있다.

$$P(C_j) = \frac{n_{c_j}}{n}$$

$P(C_j)$는 모든 학습 문서들의 수(n)와 국사 분야에 속하는 학습 문서들의 수(n_{cj})의 비

율이다. 본 프로젝트에서는 이 값을 0.5로 동등하게 적용하였다. Naive Bayesian Classifier는 $P(d|C_j)$의 계산을 단순하게 하기 위해, 문서 내에 존재하는 모든 키워드들이 서로 독립적인 관계라고 가정한다. 이러한 가정을 따르면 $P(d|C_j)$는 다음과 같은 식으로 변형될 수 있다.

$$P(d|C_j) = \sum_{i=1}^{n} \log(\frac{w_i + 1}{C_j + 2})$$

여기에서 $\frac{w_i}{C_j}$는 국사분야의 학습문서 C_j에서 키워드 w_i를 포함하고 있는 문서의 비율이다. 만약, 학습데이터에 없는 새로운 키워드가 문서에서 등장하였을 경우에는 0이 곱해지게 된다. 이러한 문제를 해결하기 위해서 분모에 1을 더하고 분자에는 분류를 수행할 클래스의 개수인 2를 더하여 Laplace Smoothing을 적용한다. $P(d|C_j)$에 대한 확률을 구하고자 할 때에는 단순히 $\frac{w_i+2}{C_j+2}$의 값들을 곱하는 것으로 충분하지만, 매우 작은 수들이 연속적으로 곱해질 경우에 산술언더플로우(Arithmetic Underflow)가 발생할 수 있다. 이를 보완하기 위해서 log값을 취해 더함으로써 이 문제를 해결한다. 임의의 문서에 대해서 해당 문서가 국사분야의 문서일 때와 그렇지 않을 때의 $P(d|C_j)$를 각각 구하여 그 값이 더 높은 분야로 문서를 분류한다.

2.4.6 문서의 주제어 추출

- 주성분 분석의 개요

주성분 분석은 다차원 적인 변수들을 축소, 요약하는 차원의 단순화와 더불어 일반적으로 서로 상관되어 있는 반응변수들 간의 복잡한 구조를 분석하는데 그 목적을 두고 있다. 주성분 분석은 반응변수들을 선형 변환시켜, 주성분이라고 부르는 서로 독립적인 새로운 인공 변수들을 유도한다. 이때, 각 주성분이 보유하는 변이의 크기를 기준으로 그 중요도 순서를 생각할 수 있는데, 그들 중 첫 소수 몇 개의 주성분이 원래자료에 내재하는 전체 변이 중 가능한 많은 부분을 보유하도록 변환시킴으로서 정보의 손실을 최소화하는 차원의 축약을 기할 수 있게 된다. 결국, 주성분 분석을

이용하여 문서의 키워드들을 축소하여 정보의 손실을 최소화하면서 소수의 몇 개 단어로 문서의 내용을 표현할 수 있다. 즉, 그 문서의 주제어를 추출할 수 있다.[1]

- 주제어 추출

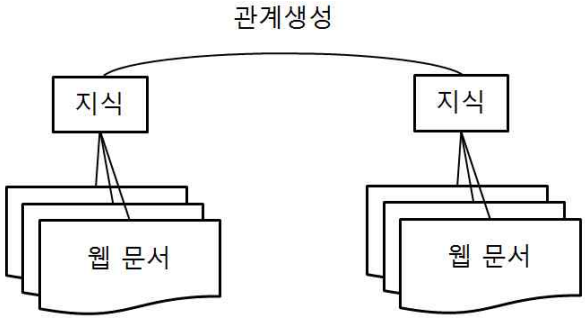
주제어 추출에는 웹문서의 구조적 특성을 이용하여 웹페이지를 Parsing하는 기법과 주성분 분석을 함께 사용하였다.

원본 제목	추출된 주제어
조선 – 위키백과, 우리 모두의 백과사전	조선
조선의 역사 – 위키백과, 우리 모두의 백과사전	조선

주제어 추출을 위해서는 가장먼저, 웹문서의 <title>태그의 내용을 Parsing하여 읽어온다. 위의 표에서 '원본 제목'으로 나타낸 항목이 이에 해당된다. 이렇게 얻은 <title>태그의 내용에서 사이트별로 일관되게 적용되는 텍스트를 삭제한다. 예를 들어, 원본 제목이 '조선 - 위키백과, 우리 모두의 백과사전'인 경우에는 ' - 위키백과, 우리 모두의 백과사전'을 삭제한다. 이렇게 했을 때, 남은 텍스트가 단일 키워드인 경우에는 주성분 분석을 수행하지 않고 단일 키워드 '조선'이 해당 문서의 주제어가 된다. 그렇지 않고 '조선의 역사'와 같이 2가지 이상의 키워드로 구성된 경우에는 먼저 '조선의 역사'에 대해서 형태소 분석을 통해 키워드를 추출한다. 그 결과로는 '조선'과 '역사' 2개의 키워드가 추출된다. 이후에는 웹 문서의 본문을 대상으로 주성분 분석을 실시한다. 주성분 분석결과에서 '조선'이 '역사'보다 해당 문서에서 더욱 중요한 변수로서 식별되면, 키워드 '조선'을 해당 문서의 주제어로서 추출한다. 결국, 제목이 '조선 - 위키백과, 우리 모두의 백과사전'인 문서와 '조선의 역사 - 위키백과, 우리 모두의 백과사전'인 문서가 같은 키워드 '조선'을 주제어로 갖는 문서로 분류되어, '조선'이라는 지식을 구성하는 하위문서로서 동일하게 저장된다.

2.4.7 지식관계 생성

- 지식관계 생성의 개요



본 프로젝트에서는 다수의 웹 문서들로 구성된 지식 간의 관계를 생성하는 방법으로서, 두 가지 지식을 구성하는 웹 문서들의 유사도를 측정하여 관계를 생성한다.

지식관계 생성을 위한 과정은 크게 3가지로 나눌 수 있다. 그 첫 단계를 각각의 키워드에 대한 TFIDF값을 계산하는 것으로, 문자 형태로 표현된 키워드를 유사도 측정을 위해서 수치값으로 변환하는 작업이다. 두 번째 단계는 VSM(Vector Space Model)을 생성하는 것인데, 계산된 TFIDF값을 이용해 n개 키워드를 보유하고 있는 문서를 n차원 벡터로서 표현하는 모델을 생성하는 것이다. 마지막 단계는 생성된 두 VSM간의 유사도 측정이다. 유사도 측정은 n차원 벡터간에 Cosine Similarity를 계산하여 유사도 점수가 일정 값 이상일 때, 두 VSM이 유사한 것으로 판단한다.

- TFIDF 산출

$$tf = f(t,d)$$

문서 d에서 단어 t의 총 빈도 f를 f(t,d)라 할 때, TF 는 단순히 해당 문서에서 키워드의 총 빈도수이다.

$$idf = \log\left(\frac{D}{df}\right)$$

IDF는 역문서 빈도로서, 전체 문서의 수 D를 해당 단어를 포함한 문서의 빈도 df로 나눈 뒤 로그를 취한 것이다. 여기에서 정적인 지식지도와는 달리 동적으로 지식지도를 작성함에 따른 차이점이 발생한다. 한정된 개수의 문서들을 대상으로 하여 TFIDF를 구할 때는 전체 문서의 개수가 고정적이므로 큰 문제가 없다. 그러나 웹상의 문서들을 대상으로 동적인 지식지도를 작성할 때는 전체 문서의 개수가 고정적이지 않고 사실상 무한 히 증가할 것이다. 그래서 문서의 개수가 증가할수록 DF(Document Frequency)를 구할 때의 비용이 증가한다. 또한, TFIDF를 사용하는 의도는 전체 문서에서 드물게 나타나고 특정 문서에서 빈번하게 나타나는 키워드에 대해서 특정 분야의 키워드로서 높은 가중치를 부여하기 위함이다. 그러나 본 프로젝트에서는 Naive Bayesian Classifier에 의해 국사분야로 분류된 문서들만을 대상으로 지식지도를 작성하므로, 실제로 국사분야의 키워드라 할지라도 일반적인 키워드와 같이 낮은 TFIDF값이 부여될 수 있다. 이러한 문제를 해결하기 위해서 DF를 구할 때는 국사분야의 문서를 대상으로 하지 않고, 국사분야가 아닌 별도의 문서들로 구성된 문서집합을 대상으로 하여 DF값을 구한다.

$$tfidf = tf \times idf$$

구해진 tf와 idf를 곱하여 tfidf를 계산한다. 이렇게 구해진 TFIDF값은 유사도를 비교하려는 두 문서간의 길이에 대한 차이를 고려하지 않은 것이다. 이 부분을 보완하기 위해서 벡터를 길이가 1인 단위벡터로 만들어 정규화 한다.

$$||tfidf||_2 = \sqrt{\sum_{i=1}^{n} tfidf_i^2}$$

길이 정규화를 하기 위해서는 각각의 키워드들에 대한 TFIDF를 제곱해서 하나로 합한다. 그리고 그 값에 루트를 씌운다. 이 값으로 전체 TFIDF값을 개별적으로 나누어서 길이를 정규화 한다.

- VSM(Vector Space Model)의 생성

벡터\키워드	Keyword Index1	Keyword Index2	Keyword Index3	Keyword Index4
V1	TFIDF	TFIDF	TFIDF	TFIDF
V2	TFIDF	TFIDF	TFIDF	TFIDF

4개의 키워드가 등장하는 2개의 문서에 대한 VSM(Vector Space Model)은 위의 표와 같은 구조로 구성된다. 유사도를 비교하기 위한 2개의 문서는 4차원 벡터 V1과 V2로 대응된다. 문서를 구성하는 키워드들은 Keyword Index로 식별되며, TFIDF는 해당 키워드가 문서 V1과 V2에서 갖는 TFIDF값을 의미한다. 즉, VSM은 Keyword Index에 대응되는 해당 문서의 TFIDF값들을 취합한 것이다.

- 유사도 측정

Cosine Similarity는 내적 공간에서 두 벡터 간 각도의 코사인 값을 이용하여 벡터 간의 유사도를 계산한다. 이때 비교되는 벡터가 몇 차원이든지 상관없이 적용될 수 있기 때문에 정보 검색 및 텍스트 마이닝 분야에서 문서 간의 유사도를 측정하는 용도로 자주 사용된다. 아래 공식은 본 프로젝트에서 유사도 측정을 위해 적용한 Cosine Similarity공식이다.

$$\cos(\theta) = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n}(A_i)^2} \times \sqrt{\sum_{i=1}^{n}(B_i)^2}}$$

여기에서 Ai는 문서 d1으로부터 만들어진 벡터 공간 모델에서 i번째 키워드에 대한 TFIDF이다. Bi는 문서 d2로 부터 만들어진 벡터 공간 모델에서 i번째 키워드에 대한 TFIDF이다. 따라서 $\sum_{i=1}^{n} A_i \times B_i$는 두 벡터간의 TFIDF를 전부 곱해서 합산한 것으로 벡터의 내적을 의미한다. $\sqrt{\sum_{i=1}^{n}(A_i)^2}$와 $\sqrt{\sum_{i=1}^{n}(B_i)^2}$는 각각 문서 d1과 d2에 대한 단위 벡터들의 곱을 말하는 것으로, 해당 벡터에 존재하는 TFIDF를 제곱해서 합산한 값에 루트를 씌운 것이다. 최종적으로 유사도 값은 벡터의 내적에 단위 벡터들의 곱을 나누어 계산한다.

2.5 현실적 제한 요소 및 그 해결 방안

2.5.1 적용된 알고리즘의 성능

- 문서의 이진분류

  본 프로젝트에 적용된 나이브 베이지안 분류기의 성능을 테스트하기 위해서 1000개의 국사분야 문서와 1000개의 국사분야가 아닌 문서를 수집하여 총 2000개의 테스트 문서로 구성된 테스트 환경을 구축하였다. 분류기의 성능을 테스트해 본 결과 13%의 문서가 오분류 되어 분류기는 87%의 정확도를 보였다. 국사분야가 아닌 문서 중에 국사분야로 오분류 된 문서는 주로 우리나라의 지역에 관한 설명을 담은 문서였다.

  해결방안

  -> 분류기를 학습할 때, 지역에 관한 문서를 국사분야가 아닌 학습데이터에 포함하여 분류기를 재학습 하거나, 문서 분류기에 적용된 알고리즘을 SVM(Support Vector Machine)과 같은 최신 알고리즘으로 교체함으로써 분류기의 성능을 향상시킬 수 있을 것으로 기대된다.

- 주제어 추출

  현재 적용된 주제어 추출 방식은 웹문서를 직접 Parsing하는 방법과 주성분분석을 이용하여 본문에서 주제어를 추출하는 방법을 함께 사용하였는데, 이러한 이유로 Seed로서 새로운 사이트를 추가하기 위해서는 해당 사이트의 Parsing 규칙을 Crawler에 수동으로 추가할 필요가 있다.

  해결방안

  -> 주성분분석을 보완할 수 있는 알고리즘을 추가로 적용하여 주제어 추출 성능을 향상시킴으로써 웹문서를 직접 Parsing하는 기존의 방법을 대체하거나, LDA(Latent Dirichlet Allocation)와 같은 알고리즘에서 문서를 모델링하는 방식을 본 따서 문서를 한 가지 키워드로 분류하는 것이 아니라, 다수의 키워드로 중복해서 분류하는 방식으로 문서에 대한 접근을 달리해서 주제어 추출의 성능을 개선할 수 있을 것으로 기대된다.

- 유사도 측정

유사도 측정에 사용된 알고리즘의 정확도는 프로젝트 기획 시에 기대했던 수준을 충족하였지만, n개의 키워드가 존재할 때 새로운 키워드가 등장하면 유사도 측정을 위한 계산을 n번 해야 하므로 유사도를 계산할 때 소모되는 시간이 지나치게 소모되는 문제가 있다.

해결방안

-> 유사도 측정을 위해 주성분분석을 적용하여 VSM(Vector Space Model)의 차원을 축소하거나, 유사도 비교 시에 모든 키워드 간에 유사도를 비교하지 않고 지식지도 상에서 연관도가 높은 키워드 집단에서 대표되는 키워드와 우선적으로 유사도를 측정하여 유사도 측정횟수를 줄이는 방식을 적용하여 유사도 측정에 소모되는 시간을 줄일 수 있다.

- 팀원 간 역할분담

팀원 간의 개인 역량차이로 인해서 소수의 특정 팀원이 대부분의 프로그래밍 작업을 맡아서 수행할 수밖에 없었으며, 이로 인해 프로젝트가 진행될수록 팀원 간 역량 차이와 작업 배분의 비효율이 점점 심화되었다.

해결방안

-> 현재 Python으로 구현된 프로젝트를 Java로 Porting하고 일부 기능을 개선하는 프로젝트를 팀원들 간에 진행하고 있는데, 이때 알고리즘 및 네트워크 프로그래밍에 대한 지식을 팀원들과 공유하여 상호 간에 미흡한 부분을 보완할 수 있을 것으로 기대된다.

2.6 기대효과 및 활용방안

- 프로젝트 기대효과

본 프로젝트가 성공 적으로 완료되면 그동안 어떠한 정보를 얻을 때 관련 지식을 추가 검색해야 하는 불편함이 줄어들 것이다. 또한 연관된 정보를 제공하여 사용자가 효율적으로 학습 정보를 습득할 수 있으며 지식지도 형태로 연관 지식에 대한 정보가 함께 제공되기 때문에 사용자의 흥미를 이끌어낼 수 있다.

현재의 프로젝트에서는 단순히 키워드들 간의 관계를 Cosine Similarity을 통하여 유사도 점수에 따라 연결해주는 방법을 사용하고 있지만 추후에 키워드 간의 관계를 설명해주는 주석을 제공한다면 사용자가 학습할 때 쉽게 이해할 수 있을 것이다.

그리고 현재는 단순히 검색서비스와 지식지도를 제공하고 있지만 사용자들이 기존에 제공되는 지식지도를 사용자에게 맞게 편집하여 개인화된 지식지도를 이용할 수 있다면 나만의 지도를 가질 수 있는 장점이 생길 것이다.

- 활용방안

교육부의 방침에 따라 한국인이 가져야 할 한국사에 대한 기본 소양을 갖추도록한다는 취지로 한국사 과목이 필수과목으로 선정되었다. 그래서 수학 능력 시험을 공부하는 고등학생들에게 본 프로젝트 서비스된다면 한국사를 공부하는데 어려움과 거부감을 줄여 상당한 기대효과를 낳을 수 있다.

2.7 UI

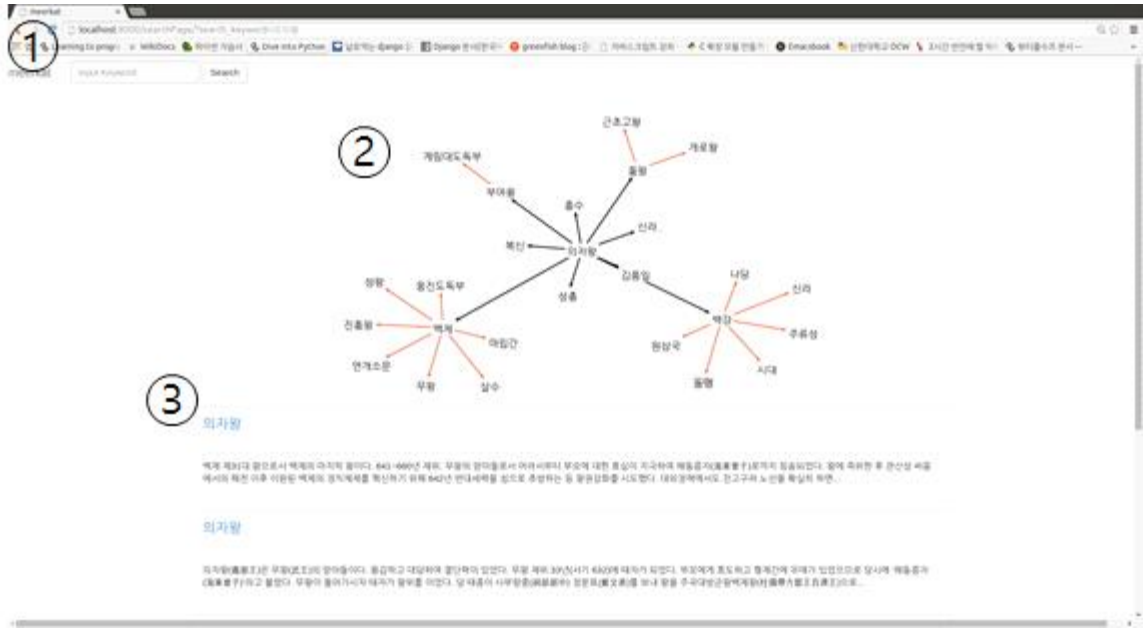
2.7.1 메인 화면

번호	기능
1	워드클라우드 형태의 키워드들을 클릭을 하면, 해당 키워드의 검색 결과 페이지로 이동하게 된다.
2	국사 분야 지식을 검색할 수 있다.

① 워드클라우드 키워드들은 각 키워드마다 연관관계 키워드가 많이 연결된 키워드들로써 키워드의 크기에 따라 연관되어지는 키워드가 많은 것이며, 그것을 크기순으로 표현하였다.

② 검색창에 국사 분야 키워드를 입력을 하면 결과 페이지로 이동된다.

2.7.2 검색 결과



번호	기능
1	메인 페이지로 이동할 수 있는 meerkat로고와 검색창으로 국사 분야 지식을 검색할 수 있다.
2	“의자왕” 이라는 키워드로 검색하였을 때 연관되어지는 키워드들 간의 연결된 지식지도이다.
3	수집한 웹문서들의 제목과 본문내용을 간단하게 보여주며 제목을 클릭하면 해당 url로 이동한다.

① 상단에 팀 프로젝트 이름인 meerkat을 로고로 사용하였고 메인페이지의 검색창과 같은 텍스트박스를 사용하였다.

② "의자왕"이라는 키워드로 검색하였을 때 "의자왕"키워드와 직접적으로 연관되는 키워드는 검정색 선으로 연결되어있고 주황색 선으로 연결된 키워드는 간접적으로 연관되어지는 키워드로써 각 키워드는 클릭하게 되면 해당 키워드의 검색결과 페이지로 이동하게 된다.

③ 키워드에 해당하는 추출된 본문내용을 지식지도 밑으로 삽입하였다. 본문 내용이 일정 개수 초과 시 다음 페이지 번호로 넘어가게 된다.

2.7.3 유사 키워드 추천

번호	기능
1	“의자왕”을 “의재왕”으로 검색한 결과이다. 이 페이지는 사용자가 검색한 키워드를 대신해 유사키워드를 추천해준다.

① 리벤슈타인거리를 이용하여 "의재왕"이란 단어를 편집거리가 가장 적은 다른 키워드들로 추천하여 보여주는 페이지이다.

3. 자기 평가

3.1 프로젝트 평가

평가부문	배점	평 가 항 목	평가점수	부문점수	총점
프로젝트 기획 (15점)	5	프로젝트 이해도	4	13	80
	5	추진전략	4		
	5	독창성	5		
기술 및 기능 (40점)	10	시스템 구조	9	32	
	10	비즈니스 로직	8		
	10	네트워킹	7		
	10	UI구현	8		
성능 및 품질 (30점)	10	프로젝트 완성도	7	26	
	10	사용성	9		
	10	신뢰성	10		
프로젝트관리 (15점)	5	일정계획	5	9	
	5	협업관리	2		
	5	팀워크	2		

3.2 개별 평가

성명	동승일
프로젝트 수행 소감	한마디로 많이 아쉬웠던 프로젝트였다. 3년동안 공부해온 것들을 집약적으로 보여주는 것이 졸업작품 프로젝트인데 나의기여도는 매우 낮았다. 그럴 수밖에 없는 것이 일단 아는 것이 많이 부족했다. 해서 프로젝트의 핵심적인 구현이나 코딩부분보다 자료수집과 문서작업에 동원되었다. 프로젝트 레벨에 미치치 못하는 역량탓에 팀원들의 발을 붙잡은게 아닌지 생각하며 이 페이지를 빌어 미안하다고 마음으로부터 깊이 감사하다는 말을 전하고 싶다. 앞으로 취업을 위해 역량강화에 정진해야겠다는 강한 동기부여가 됐다.

성명	정해덕
프로젝트 수행 소감	약 7개월 동안 진행되었던 프로젝트가 끝이나 한편으로는 후련하고 한편으로는 아쉬움이 남는 시간이었다. 처음 목표했던 결과보다는 중간에 수정이 많이 되어 부족한 면이 존재하는 프로젝트로 남았는데, 추후에 프로젝트를 파이썬 언어에서 자바언어로 포팅 하여 지금보다 더욱더 완성도 높은 결과물을 도출할 예정이다. 프로젝트를 진행하는 동안 서로의 역량 차이가 커 역할분담이 뜻대로 안되었다. 소수의 팀원이 많은 부담을 지며 프로젝트를 진행하게 되어서 아쉬웠다.

성명	이상협
프로젝트 수행 소감	팀장으로서 모든 팀원들이 함께 참여해서 열심히 진행했었더라면 더 많은 것을 배우고 결과물도 좋았을 것이라는 아쉬움이 가장 많이 남았다. 하지만, 한편으로는 크고 작은 결정에 있어서 내가 독단적인 생각에 사로잡혀 올바르지 못한 판단을 했을 때에도 불평불만 하지 않고 끝까지 프로젝트를 함께해준 팀원들에게 고마웠다. 졸업작품으로써 프로젝트를 완료한 후에도 이 프로그램을 계속 개발하면서 스터디 하는 것으로 팀원들과 약속하였는데, 지금까지 팀원들에게 잘 못 해왔던 것을 바로잡고 무엇보다 팀원들이 재미있게 프로그래밍 할 수 있는 기회가 되도록 만들기 위해 노력할 것이다.

성명	김학성
프로젝트 수행 소감	학기 초부터 지금까지 진행하였던 졸업작품이라 말도 많고 탈도 많았지만 그만큼 애착이 생긴다. 처음 목표와는 많이 다르게 프로젝트 내용이 조금 변경되고 역할분담도 많이 달라져왔지만 나름대로 끝까지 포기하지 않은 팀원들과 따라갈 수 있게 길을 잡아준 팀장에게 정말 고마움을 느낀다. 프로젝트를 진행하면서 프로그래밍 학습 이외에 정말 많은 기술과 경험을 쌓게 되어 정말 값진 시간들이었다고 생각한다. 이 프로젝트를 졸업작품 이후로도 팀원들과 같이 계속 발전시키도록 할 것이며, 더 정진할 수 있도록 노력할 것이다.

4. 참고문헌

번호	종류	제목	출처	발행년도	저자
1	연구보고서	지식지도 작성을 위한 기초 연구	국가과학기술정보센터	2007	이광희
2	논문	주성분 분석을 이용한 문서 주제어 추출	국회도서관	2002	이창범 외 4명
3	논문	Naive Bayes 문서분류기를 위한 점진적 학습 모델 연구	국가과학기술정보센터	2001	김제욱 외 2명
4	서적	Building Machine Learning Systems with Python	에이콘	20014	윌리 리커트 외 1명
5	웹	http://blog.hannal.net/01-python_django_lecture/			
6	웹	http://opentutorials.org/course/477			
7	웹	http://ra2kstar.tistory.com/86			
8	웹	http://csstudy.wordpress.com/2014/03/04/naive-bayes-classifier-나이브-베이지안-분류/			
9	웹	http://unlimitedpower.tistory.com/entry/NLP-Naive-Bayesian-Classification나이브-베이즈-분류			