**캡스톤 디자인 I**

**종합설계 프로젝트**

|  |  |
| --- | --- |
| 프로젝트 명 | *COPYDET* |
| 팀 명 | *Detective* |
| 문서 제목 | 결과보고서 |

|  |  |
| --- | --- |
| **Version** | 0 |
| **Date** | 27 |

|  |  |
| --- | --- |
| **팀원** | 강 유진 (조장) |
| 김 승환 |
| 이 아르미 |
| 이 현영 |
| 전 승철 |
| **지도교수** | 강 승식 교수님 |

|  |
| --- |
| **CONFIDENTIALITY/SECURITY WARNING**  이 문서에 포함되어 있는 정보는 국민대학교 전자정보통신대학 컴퓨터공학부 및 컴퓨터공학부 개설 교과목 캡스톤 디자인I 수강 학생 중 프로젝트 “COPYDET”를 수행하는 팀 “Detective”의 팀원들의 자산입니다. 국민대학교 컴퓨터공학부 및 팀 “Detective”의 팀원들의 서면 허락없이 사용되거나, 재가공 될 수 없습니다. |

**문서 정보 / 수정 내역**

|  |  |
| --- | --- |
| **Filename** | 결과보고서- COPYDET.pdf |
| **원안작성자** | 강유진, 김승환, 이아르미, 이현영, 전승철 |
| **수정작업자** | 강유진, 김승환, 이아르미, 이현영, 전승철 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 수정날짜 | 대표수정자 | Revision | 추가/수정 항목 | 내 용 |
| 2014-05-20 | 홍길동 | 1.0 | 최초 작성 |  |
| 2014-05-22 | 홍길동 | 2.0 | 내용 수정 | 수정된 연구내용 추가 |
| 2014-05-24 | 홍길동 | 2.1 | 내용 수정 | 향후 추진 계획 수정 |
| 2014-05-25 |  | 2.2 |  |  |
| 2014-05-26 |  | 2.3 |  |  |
| 2014-05-27 |  | 3.0 |  |  |
|  |  |  |  |  |

**목 차**

[1 개요 4](#_Toc388896660)

[1.1 프로젝트 개요 4](#_Toc388896661)

[1.2 추진 배경 및 필요성 5](#_Toc388896662)

[1.2.1 프로젝트의 추진배경 5](#_Toc388896663)

[1.2.2 프로젝트의 필요성 6](#_Toc388896664)

1.2.2.1 기존 표절의 검사방법

1.2.2.2 프로젝트 진행의 필요성

[2 개발 내용 및 결과물 10](#_Toc388896665)

[2.1 목표 10](#_Toc388896666)

[2.2 연구/개발 내용 및 결과물 11](#_Toc388896667)

[2.2.1 연구 내용 11](#_Toc388896668)

2.2.1.1 연구내용

2.2.1.2. 개발에 쓰인 알고리즘 및 기법

(1) TF-IDF

(2) IR Vector Space Model : Cosine Similarity

(3) Stanford POS(Part ?Of-Speech) tagger

2.2.1.3 개발환경 구성

(1) 운영체제

(2) 사용 프로그래밍 언어

(3) 사용 모듈 라이브러리

[2.2.2 개발 내용 16](#_Toc388896669)

2.2.2.1 Source Retrieval

(1) Source Retrieval 개발 개요

(2) Source Retrieval 개발 수행사항

2.2.2.2 Text Alignment

(1) Text Alignment 개발 개요

(2) Text Alignment 알고리즘

(3) Text Alignment 알고리즘 구현

2.2.2.3 User Interface

[2.2.3 시스템 기능 및 구조 설계도 33](#_Toc388896670)

[2.2.4 현실적 제한 요소 및 그 해결 방안 34](#_Toc388896671)

[2.2.5 결과물 목록 35](#_Toc388896672)

[3 자기평가 36](#_Toc388896673)

[3.1 기대효과 및 활용방안 36](#_Toc388896674)

[3.2 자기평가 37](#_Toc388896675)

[4 참고 문헌 38](#_Toc388896676)

[5 부록 40](#_Toc388896677)

[5.1 사용자 매뉴얼 40](#_Toc388896678)

[5.2 운영자 매뉴얼 40](#_Toc388896679)

[5.3 배포 가이드 40](#_Toc388896680)

[5.4 XXX 매뉴얼 40](#_Toc388896681)

[5.5 XXX에 대한 기술 문서 40](#_Toc388896682)

# 개요

## 프로젝트 개요

최근 몇 년 사이에 '연구윤리','연구부정'및 '표절'의 문제가 학계나 문화계를 포함한 사회 각 부문에서 하나의 커다란 사회 문제로 등장하였다.







그림 1.1-1 신문기사

표절이란 다른 사람이 창작한 저작물의 일부 또는 전부를 도용하여 사용하여 자신의 창작물인 것처럼 발표하는 것을 말한다. 한국에서는 교수 출신 공직자들의 논문 표절이 사회적 이슈가 되면서 각 대학이나 학회 별로 표절 심사 기준을 마련하고 있다. 그러나 이러한 노력에도 불구하고 다양한 분야에서 다양한 유형으로 표절이 행해지고 있다.

특히 스마트기기가 확산되면서 정보를 접하는 것이 쉬워짐에 따라 인터넷 상의 글을 베껴 쓰거나 여러 글을 편집하여 쓰는 등 표절의 유형은 다양해지고 있다. 뿐만 아니라 정보 접근이 쉬어짐에 따라 의도치 않게 표절이 이루어지기도 한다. 본 프로젝트는 예술, 학문 등 여러 분야에 만연한 표절의 실태를 파악하고 기존의 표절 검사 프로그램의 문제점을 분석하여 표절을 가려내는 시스템을 개발한다.

그리고 팀 프로젝트를 통해 협동심, 개인 업무 부담, 프로젝트 일정조정 등, 팀 프로젝트를 통한 개인프로젝트가 아닌 여럿이서 하는 프로젝트 경험을 통하여 필요한 다양한 요소들을 습득한다.

## 추진 배경 및 필요성

### 프로젝트의 추진배경

얼마 전, 교육부 장관의 논문 표절 의혹이 불거졌다. 문화계•정치계•연예계 등 다양한 분야에서 표절 문제는 끊임없이 흘러나오고 있다. 이는 비단 개인의 문제는 아닐 것이다. 표절은 심각한 정보 윤리 위반이며 우리나라뿐만 아니라 세계적으로 해결해야 하는 문제이다.

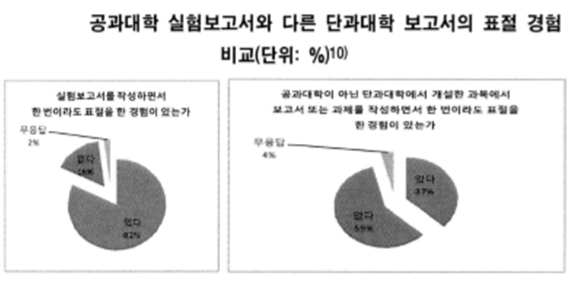


그림 1.2.1-1 표절경험 비교

대학에서 가장 빈번하게 발생하는 비윤리적인 행동 중 한 가지가 표절(Plagiarism)이다. 인터넷의 발달로 학생들이 과제를 제출하거나 보고서를 제출할 때 웹사이트에서 쉽게 자료를 복사해서 제출할 수 있고 실제로 많은 학생들이 이러한 방식으로 과제를 하고 있기 때문에 학생들은 표절이 비윤리적인 행위라는 인식 없이 행하고 있다.

서울대 공대생 100명을 대상으로 설문 조사를 한 결과 “실험 보고서 작성 시 표절을 한 번 이상 한 경험이 있다.”고 답한 학생이 82%, “공대가 아닌 다른 단과 대학의 보고서 작성 시 표절을 한 번 이상 한 경험이 있다”고 답한 학생이 37%였다. 또 현재의 표절 대응책의 효과가 거의 없거나 전혀 없다고 한 학생이 77%였다. 이를 바탕으로 다른 학문 분야의 과제물에서보다는 공대에서 표절이 더 빈번하다는 것을 알 수 있다.

외국의 경우 표절을 연구 부정으로 간주하여 엄격히 금지하며, 표절에 대한 교육을 실시하고 있다. 이러한 표절행위는 학문의 창의성을 감소시키며, 연구의 정직성을 훼손시키게 된다.

정보화 시대에서 인터넷은 항상 빠르고 쉽게 만족할만한 정보를 접근할 수 있는 개념의 장으로 인식되어왔다. 하지만 이와는 반대로 신종 학문 범죄가 일어나는 장으로 변질 되고 있으며, 이렇게 얻어진 정보가 무분별하게 도용되는 사례가 증가하고 있다. 이처럼 인터넷과 같이 전자 매체가 발달한 현재에 수많은 문서를 사람이 직접 비교하여 표절여부를 판별하는 것은 사실상 불가능하다. 따라서 컴퓨터를 이용해 문서의 특징을 파악하고 이를 토대로 표절 검사를 수행하는 시스템이 필요하다.

### 프로젝트의 필요성

#### 1.2.2.1 기존 표절의 검사방법

표절검사에서 주된 관심사는 표절에 사용한 교재, 신문, 잡지, 혹은 인터넷의 원 문서를 확인하여 표절여부를 판정하는 것이다. 그러나 표절검사를 위안 통합 데이터베이스 시스템이 구축되어 있지 않기 때문에 문제가 발생했을 때 수작업으로 진행될 수밖에 없는 것이 현실이다. 또한 과제물의 표절검사는 교수가 수작업을 통해 내적 표절을 확인하고 있으나, 그 정확도와 신뢰도는 현저히 낮으며, 외적 표절에 대한 검사는 현실적으로 불가능한 실정이다. 따라서 상당수의 학생들은 인터넷 검색을 통해 무분별하게 표절을 일삼고 있다.

표절 여부를 검사하는 방법에 있어 두 가지 다른 접근 방법이 제시되고 있다. 특징적인 단어에 초점을 두는 지문(fingerprint) 검사 방법과 전체 구성의 흐름에 초점을 두는 구조 기반 검사 기법이 있다. 두 가지 검사 방법을 설명하기에 앞서 일반 문서 표절 기법 대해 살펴보고, 이 방법들에 대응하는 표절 여부 검사 방법을 제시하도록 하겠다.

대체로 표절을 할 때에는 원본의 내용을 완벽히 이해하여 새로운 문서로 만들어내는 것이 아니라 짧은 시간 내에 일부분을 편집하여 사본을 만들어내게 된다. 그렇기 때문에 문서의 원본과 사본을 비교해 보면 다음과 같은 특징을 가진다.

**- 일부 단락이 삭제되거나 순서가 재배치되는 경우, 또는 원본에는 없는 단락이 삽입되어 있다.**

**- 일부 문장을 편집하여 새로이 작성하였지만 주제어는 그대로 사용한다.**

**- 문서의 구조나 단락의 구조가 동일하다.**

**- 틀린 철자를 그대로 사용한다.**

**- 문서에서 일부 단어를 의미가 동일한 단어로 교체하였다.**

문장이나 문단의 주제와 내용은 동일한데 문장의 편집이나, 단어를 유사한 다른 단어로 교체하는 단순한 편집 과정을 거쳐 표절이 이루어진다. 또 두 종류 이상의 문서를 섞어 하나의 문서로 만드는 경우는 앞 뒤 단락의 흐름이 자연스럽지 못하다.

특히 전자 매체가 발달된 현재에는 사람이 직접 문서의 표절의 여부를 확인하는 것은 불가능한 일이다. 그렇기 때문에 컴퓨터를 이용해 문서의 특징을 파악하고 이를 토대로 표절 검사를 확인하는 작업이 필요하다. 문서는 선형 구조를 이루고 있기는 하나 문서의 크기에 따라 선형 구조의 길이가 가변적이기 때문에 구조적인 특징보다 통계적인 특징을 추출하여 이를 토대로 표절 검사한다. 즐겨 사용하는 단어나 문서 내에서 사용된 단어의 빈도 수, 단어, 문장, 문단의 평균길이 그리고 느낌표나 물음표 등의 부호 사용 횟수 등을 비교하여 표절 유무를 검사하는데 이러한 방법을 지문법(fingerprint)라고 한다.

|  |  |  |  |
| --- | --- | --- | --- |
| 시스템 | 검사대상 | 검사방법 | 비고 |
| Plagiarism.org | 일반문서 | 비공개 | 온라인 운영 및 DB |
| IntergirGuard | 일반문서 | 비공개 | 온라인 유료 사이트 |
| EVE2 | 일반문서 | 비공개 | 온라인 및 유사 문서검색 |
| CopyCatch | 일반문서 | 공통어휘 및 빈도수검사 | 오프라인 시스템 |
| WordCheck | 일반문서 | 단어 사용 횟수 검사 | 오프라인 시스템 및 DB |
| COPS | 일반문서 | 문장 일치 여부 검사 | 오프라인 시스템 |
| SCAM | 일반문서 | 단어 일치 여부 검사 | 단어를 DB에 저장 |
| 교수클럽 | 일반문서 | 단어, 특수문자, 공식 등의 빈도수검사 | 온라인 유료 사이트 |
| SIM | 소스코드 | 토큰 참조 횟수 비교 | - |
| Siff | 소스코드 | 50개 대표 문자 추출 비교 | - |

표 1.2.2.1-1 지문법을 사용하는 표절검사 시스템

지문법을 이용하면 문서나 프로그램의 지문은 추출해낼 수 있지만 구조적인 분석은 어려워진다. 문서와는 달리 프로그램은 제어 흐름의 변경이 어려우며, 프로그래밍 언어의 문법이 정해져 있기 때문에 구조적인 특성이 잘 나타난다. 구조기반(Structure-based) 표절 검사 방법은 문서의 표절 검사보다 제어흐름을 가지고 있는 프로그램 소스 코드의 표절 검사에 많이 사용된다.

|  |  |  |  |
| --- | --- | --- | --- |
| 시스템 | 검사대상 | 검사방법 | 비고 |
| CHECK | LATEX 문서 | 문서의 특성을 트리로 구성 | 지문법을 일부 사용 |
| Plague | 프로그램 코드 | Longest Common Subsequence |  |
| YAP | 프로그램 코드 | 스트링 매칭 방법 적용 |  |
| YAP3 | 프로그램 코드 | Karp-Rabin, Greedy-String-Tiling | 영어로 된 과제 검사 |
| MOSS | 프로그램 코드 | 스트링 매칭 방법 적용 | 온라인 시스템 |
| JPlag | 프로그램 코드 | Karp-Rabin, Greedy-String-Tiling | 온라인 시스템 |
| Code Match | 프로그램 코드 | 5가지 매칭 알고리즘 | 상용화 소프트웨어 |

표 1.2.2.1-2 구조기반 검사방법을 사용하는 표절검사 시스템

문서의 표절 검사를 위해 많이 사용되는 지문법은 문서의 길이에 영향을 받지 않는다는 장점을 가진다. 또한 문서에서 지문을 뽑아내기만 하면 이 지문들끼리 비교를 통해 표절 여부를 판단하기 때문에 시간 복잡도도 낮다. 그러나 이 지문법은 부분적인 표절 여부를 찾아내기가 어렵고, 구조적인 특성을 가지는 프로그램 코드의 검사에 적용하기에는 신뢰성이 부족하다. 지문법과 구조기반 검사방법에 대한 비교는 다음의 표 1.2.2-3와 같다.

|  |  |  |
| --- | --- | --- |
| 설명 | 지문법 | 구조기반 검사 방법 |
| 문어에서 얻어내는 정보 | 유동적 | 고정적 |
| 문서의 길이 | 영향을 받지 않음 | 길이에 따라 언어내는 정보가 비례함 |
| 부분적인 표절 탐지 | 어려움 | 쉬움 |

표 1.2.2.1-3 지문법과 구조기반 검사 방법의 비교

#### 1.2.2.2 프로젝트 진행의 필요성



그림 1.2.2.2-1 표절논란 현황

유명인사의 '논문표절 논란'이 끊이지 않는 사회적 이슈로 자리 잡은 가운데 대학들에도 '연구윤리 의식'은 개선해야 할 하나의 필수과제가 되었다. 최근 많은 대학들은 학위논문 연구윤리심사 강화를 위한 교육부 지침에 따라 의식 확립을 위해 다양한 노력을 펼치고 있다. 일부 대학들은 학위논문 제출 예정자를 대상으로 '연구윤리 교육과 논문표절방지'에 대한 주제로 교육을 진행하고 있다.

한 대학교수는 "논문지도를 하다 보면 혹시나 내가 지도하는 학생이 표절의심을 받을지도 모른다는 생각에 굉장히 부담되는 것이 사실"이라며 "표절검사 서비스를 이용하여 일차적 참고도구로 표절 여부를 확인할 수 있다는 것은 그러한 부담을 덜어줄 수 있는 유용한 예방책"이라고 말했다.

표절검사 서비스는 논문 심사 시 평가도구뿐만 아니라, 실제 작성 중에 활용하며 다양한 연구영역에서 사용할 수 있는 길잡이 역할을 한다는 점에서 대학 및 연구기관 사이에서 주목을 받을 것으로 기대된다.

이뿐만 아니라 교직원 및 학생을 대상으로 한 연구윤리 교육, 논문 작성법, 주요 표절 사례 등에 대한 교육도 함께 제공하여 교내 구성원들의 연구윤리 의식 함양뿐만 아니라 학위논문 질적 제고를 위할 수 있다.

해외 표절검사 서비스의 경우 학위논문 비교자원이 부족해 국내 대학교 및 기관에서 이용하기에는 한계점이 있고, 이용횟수 및 검사시간의 제약 혹은 한글과 영어의 구조적 차이로 인한 한계가 있어 문서 작성 중에 자주적 글쓰기 도구로써 활용되기에는 불편하다. 연구 과정 중에 언제든지 표절 여부 실시간 검사를 통해서 연구자들의 자율적인 연구윤리 실천문화를 조성할 수 있다.

# 개발 내용 및 결과물

## 목표

본 프로젝트의 목표는 학교 및 여러 교육기관들과 공공기관들에서 문서 표절 여부를 쉽게 찾아낼 수 있는 프로그램을 만드는 것이다. 사용자들은 이 프로그램을 통해서 손쉽게 문서들의 표절여부를 찾아낼 수 있으며 또한 사용자가 문서표절에 소비하는 시간과 비용을 절감할 수 있다. 또한 이 프로그램을 통해 사회 전반적으로 만연해 있는 표절에 대한 경각심을 일으키고, 표절 근절에 앞장설 수 있다.

최근 들어 유명인사의 학력 위조, 학계의 논문 표절 등이 사회적 이슈로 대두됨에 따라, 표절에 대한 관심도 급증하고 있다. 특히 그 중에서도 논문, 기사 등 한글문서의 표절이 논란이 되고 있는데, 특허청에서는 표절을 “다른 사람의 저작물의 전부나 일부를 그대로, 또는 그 형태나 내용에 다소 변경을 기하여 자신의 것으로 제공 또는 제시하는 행위”라고 정의하고 있다. 이러한 문제들이 사회에서 이슈가 되듯이, 문서 간 표절여부, 즉 문서간의 유사성을 검사하는 시스템의 연구 개발이 상당히 활발히 진행이 되고 있는 상태이다. 따라서, 현재의 유사 문서 탐색 시스템에서는 유사 문서 판정의 정확도는 물론 다량의 문서에 대한 탐색 속도 또한 중요한 척도가 되고 있다. 우리는 이 프로젝트를 통하여 기존 유사 프로그램들보다 표절 검사 속도와 정확도를 향상시키고자 한다. 일반적으로 유사 문서 탐색 속도는 대상 문서의 개수에 좌우되므로 모든 검사 대상에 대해 검사를 수행하는 것은 비효율적이다. 유사 문서 검사를 하기 전에 전처리 과정을 통하여 유사 가능성이 높은 문서만을 추출한다면 유사 문서 탐색 속도의 향상을 기대할 수 있다.

본 프로젝트에서는 전처리 과정으로 의심문서와 유사 문서인 소스문서를 추출을 과정을 통하여 표절문서의 탐색시간을 줄이고 정확도를 높이고자, 수많은 문서군 사이에서 Source-Retrieval을 통해 표절이 의심되는 문서가 표절했다고 생각되는 문서들을 선별한 뒤 Text-Alignment로 각 문서간의 표절구간을 정밀하게 탐색한다.

## 연구/개발 내용 및 결과물

### 연구 내용

#### 2.2.1.1 연구내용

문서들의 표절여부를 검사를 하기 위하여 표절의심문서와 소스문서군들 간의 유사도 검색 후 유사도가 일정치 이상의 문서들 간의 표절 구간을 찾는 프로그램을 만든다. 주요 모듈로는 Source-Retrieval과 Text-Alignment가 있다. 언어 처리에 탁월한 기능을 가지고 있으며 현재 대중적으로 많이 쓰이고 있는 Python을 개발 언어로 채택하였다.

Source-Retrieval의 역할은 자세한 표절검사를 하기 전에 표절 의심문서에서 단어를 추출하여 표절 의심 문서가 웹 상의 어떤 문서를 표절하였는지 찾아주는 것이다. 웹 상에서 표절 소스 문서가 발견되면 표절 의심 문서와 쌍을 지어 text로 출력한다.

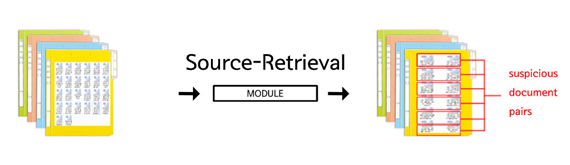


그림 2.2.1.1-1 Source-Retrieval 모듈의 역할

Text-Alignment모듈은 Source-Retrieval문서에서 넘겨준 문서 쌍의 표절률과 표절구간을 찾아주는 역할을 하는 모듈이다..

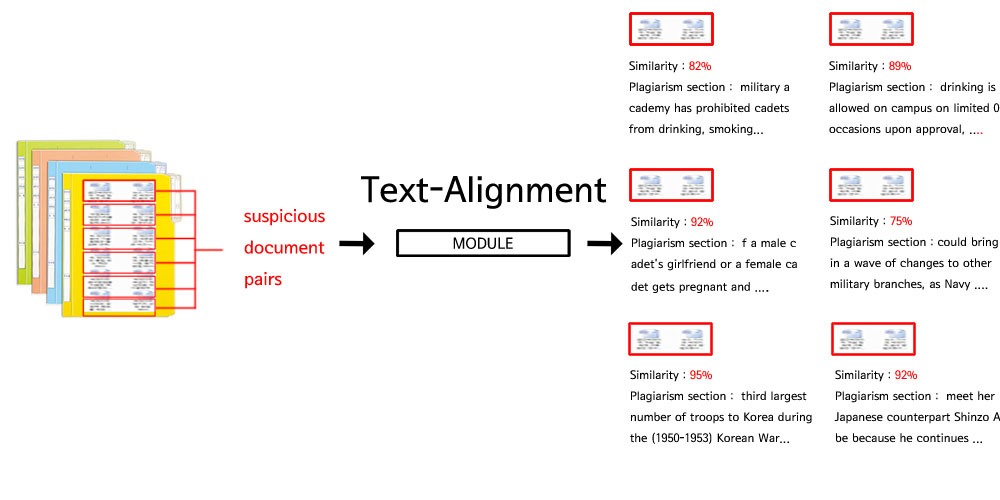


그림 2.2.1.1-2 Text-Alignment 모듈의 역할

#### 2.2.1.2 개발에 쓰인 알고리즘 및 기법

##### TF-IDF

TF(Term Frequency): 단어 빈도수는 말 그대로 "단어가 그 문서에서 나타난 횟수"를 나타낸다. 단어 빈도수는 1이 아닌 그 문서에서 나타난 횟수를 나타냄으로써, 그 문서에서 그 단어가 얼마나 중요한지를 표현 할 수 있다.

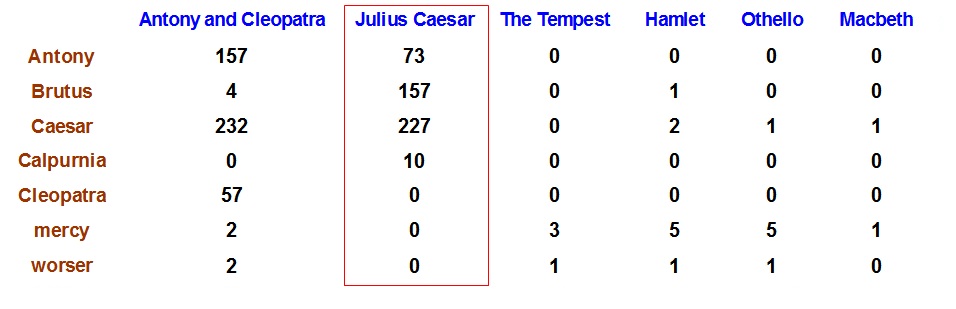


그림 2.2.1.2-1 TF의 예시

DF(Document Frequency) : DF는 문서 빈도수는 "해당 단어가 나타난 문서의 수" 이다. DF 값이 높은 단어는 많은 문서에서 나타나는 것이므로, 이 문서군에서 많이 쓰이는 단어임을 알 수 있다. 그러므로 DF가 높은 단어는 중요도가 떨어진다. 즉, 검색에서 별로 중요한 단어가 아니라는 것을 나타낸다.

IDF(Inverse Document Frequency) : 역문서 빈도수는 DF를 역수 취한 것이다. DF는 값이 클수록 중요하지 않은 단어를 나타내는 것인데, IDF의 값이 클수록 DF 값은 작아진다.

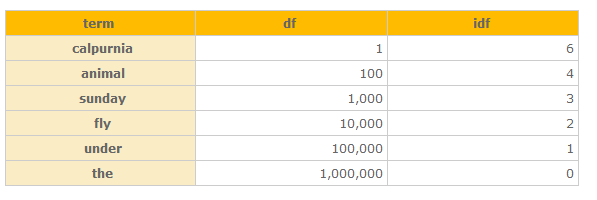


그림 2.2.1.2-2 DF와 IDF의 예시

TF-IDF : TF-IDF는 단어의 중요도를 수치로 표현 한 것으로써, TF \* IDF으로 이루어 진다.

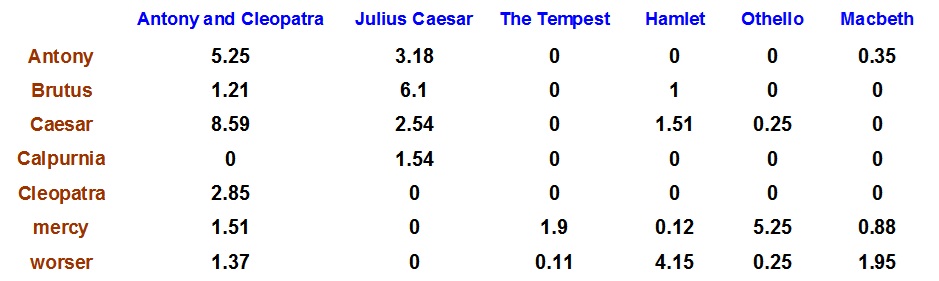


그림 2.2.1.2-3 TF-IDF 예시

##### IR Vector Space Model : Cosine Similarity

벡터 공간 모델에서 문서(document)는 벡터로 표현되며, 각각의 차원(dimension)은 개별 단어(term)에 대응된다. 만약, 문서 내에 특정 단어가 포함되어 있다면, 벡터 내에서 해당 차원은 0이 아닌 값을 갖게 된다. 단어 가중치(term weight)라고 하는 이 값을 계산하는 데에는 여러 가지 방법이 있으며, 가장 잘 알려진 방법 중의 하나는 TF-IDF 가중치를 구하는 방법이다.

벡터 공간 모델은 키워드 검색 시, 문서의 유사도 순위를 매기는 데에 응용될 수 있다. 문서 유사도 이론의 가정을 이용하여, 각각의 문서 벡터와 (쿼리를 문서들과 같은 종류의 벡터로 표현한) 쿼리 벡터 사이의 각도를 비교해서 유사도 순위를 매긴다. 코사인 값이 0인 경우는 쿼리와 문서 벡터가 직각인 경우로, 둘 간의 유사성이 전혀 없음을 의미한다. 그리고 코사인 값이 1에 가까울수록 두 문서간의 유사성이 높다는 것을 의미한다.

http://cfs8.tistory.com/original/8/tistory/2008/06/04/12/57/484612a3dee2b

그림 2.2.1.2-4 코사인 계산법

문서 d의 가중치 벡터 http://cfs7.tistory.com/original/17/tistory/2008/06/04/12/58/484612cad9e31이고http://cfs7.tistory.com/original/23/tistory/2008/06/04/12/58/484612ebb21c8이다.

여기서, tf는 문서 d에서 단어 t가 등장하는 단어 빈도(term frequency)이며 (지역 매개 변수),

http://cfs8.tistory.com/original/12/tistory/2008/06/04/13/00/48461346977a7 는 는 역 문서 빈도(inverse document frequency)이다. (전역 매개 변수)

| D |는 전체 문서 수이며, http://cfs8.tistory.com/original/17/tistory/2008/06/04/13/00/4846135fb6ec4는 은 단어 t를 포함하고 있는 문서들의 숫자이다

##### Stanford POS(Part –Of-Speech) tagger

Stanford NLP연구소에서 개발한 Python용 모듈로써 문서를 Token화 하여 각 Token마다 품사태그를 표시하는 방식이다. 태그 종류는 다음과 같다.

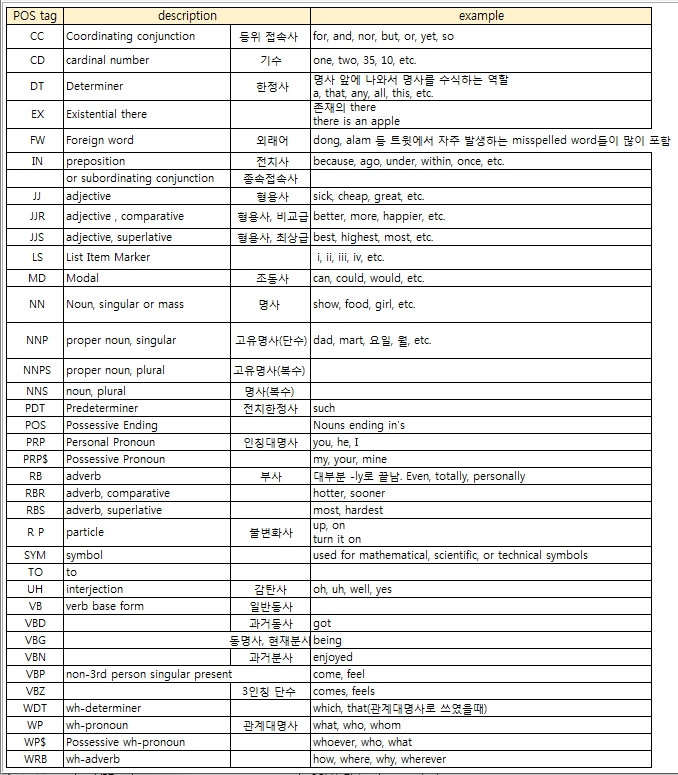


그림 2.2.1.2-5 POS tagger의 종류

#### 2.2.1.3 개발환경 구성

##### 운영체제

- Windows7 32bit/64bit

##### 사용 프로그래밍 언어

- Python : Source Retrieval과 Text-Alignment, UI를 개발하여 window system application을 제작

##### 사용 모듈 라이브러리

- NLTK (Natural Language Tool Kit) : NLTK는 자연어로 데이터를 구축하는 파이썬 프로그램을 만드는 것을 위한 최고의 플랫폼이다. 이 라이브러리는 분류, 토큰화, 형태소분석, 태킹, 파싱, 그리고 시맨틱 추론과 같은 라이브러리 제품을 제공한다. NLTK는 윈도우, 맥, OS X, 리눅스에서 사용할 수 있습니다. 그리고 무엇보다도 NLTK는 무료, 오픈소스, 커뮤니티 기반 프로젝트이다.

- NUMPY: 파이썬에서 과학연산을 위해 사용하는 기본적인 패키지이며, 강력한 N 차원 배열 객체, 유용한 선형 대수, 푸리에 변환, 및 난수 기능을 포함하고 있다.

- SCITIT LEARN : 데이터 마이닝 및 데이터 분석을 위한 간단하고 효율적인 도구, 다양한 상황에서 모두에게 접근 가능하고, 재사용 가능하다. 분석, 회귀 분석, 클러스터링, 차원축소, 모델 선택, 전처리 등의 라이브러리를 제공한다.

- wxPython : wxPython는 Python 언어를 위한 GUI 툴킷으로 쉽고 간단하게 GUI를 구성할수 있다.오픈소스로 누구나 사용할 수 있고 크로스 플랫폼을 지원하여 Windows, Unix 또는 Unix 계열, Macintosh OS X등에서 사용이 가능하다.

### 개발 내용

#### 2.2.2.1 Source Retrieval

##### Source-Retrieval 개발 개요

문서들간의 유사도를 검색하여 유서도가 일정한 수치를 넘은 문서들끼리 Text-Alignment를 하도록 TF-IDF기법으로 유사도가 높은 문서 쌍을 검출한다. Python기반으로 만들어졌다.

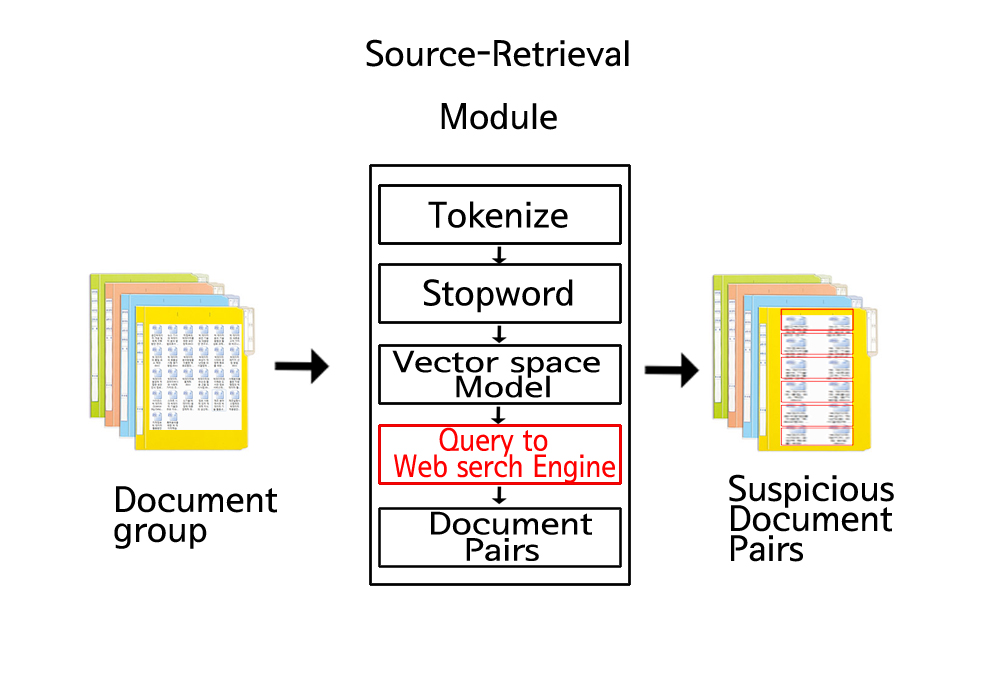


그림 2.2.2.1-1 Source-Retrival 개념도

1. **Tokenize(토큰화)** : 문서의 내용을 리스트형태로 토큰화 시킨다.
2. **POS Tagger** : 리스트의 각 token에 품사 tag를 표시한다.
3. **Stopword(불용어 제거)** : 토큰화한 리스트에서 불용어를 제거해준다.
4. **Vector space Model(유사도 추출)** : 벡터 스페이스 모델을 이용하여 문서의 유사도 순위를 매긴 후 유사도가 비슷한 문서들끼리 쌍을 묶어 준다.
5. **Query to Web Search Engine(웹 문서 비교)** : 문서에서 특정 키워드를 이용하여 Web earch Engine에 쿼리를 보낸다.
6. **Document Pairs** : Vector space Model에서 추출한 유사도로 비슷한 문서들끼리 쌍을 만들어 주고 그 이후에 Text-Alignment로 넘겨준다.

##### Source-Retrieval 개발 수행사항

1. **Web Search API**

현재 공개되어있는 Web Search API를 이용하여 표절문서인지 아닌지 판단하는 Source-Retrieval프로그램을 만든다. 공개되어 있는 Web Search API인 Char-Noir를 이용한다.

1. **불용어 처리 함수**

토큰화한 문서파일들에서 불용어들을 제거하기 위하여 만든 함수다. 기본적으로 텍스트파일에서 라인 단위로 구분을 하며 손쉽게 추가 제거가 가능하다.

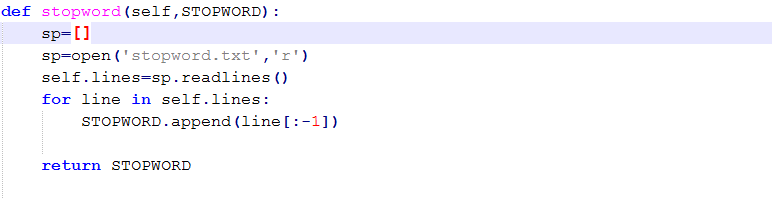


그림 2.2.2.1-2 불용어 처리 함수

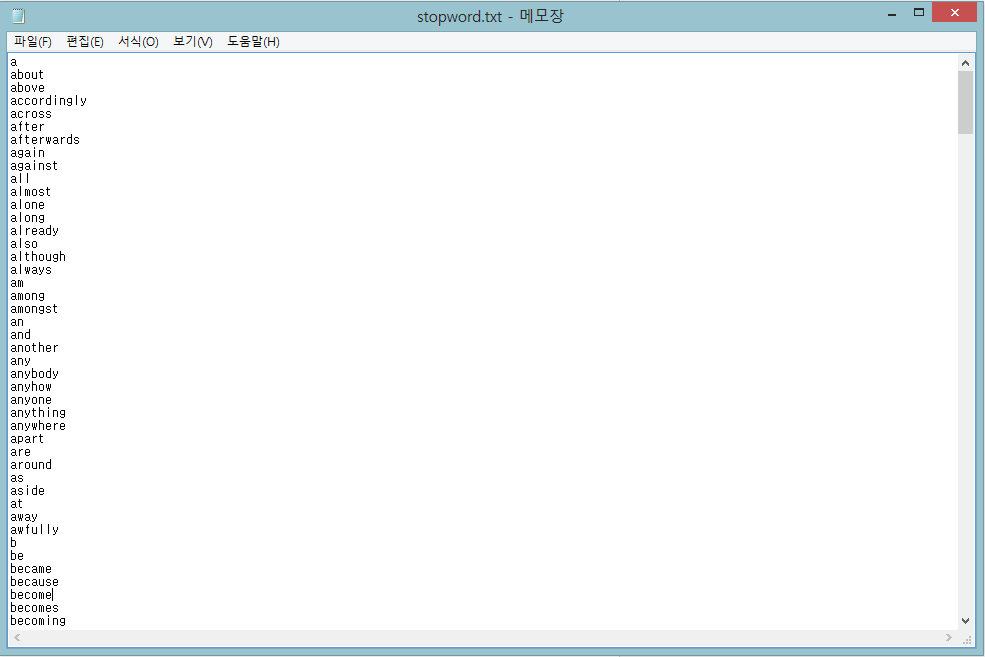


그림 2.2.2.1-3 불용어 목록

1. **토큰화 함수**

문서를 토큰화 해주는 함수. 토큰화 할 때 불용어 처리를 하여 불용어 목록에 있는 토큰들을 제거하고 리스트형식으로 만든다. 그리고 완성된 리스트에서 Stanford-PosTagger를 이용하여 각각의 토큰에 품사 tag를 달아준다.

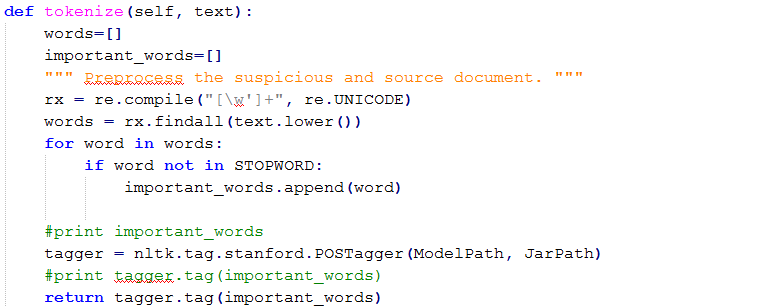


그림 2.2.2.1-4 토큰화 함수



그림 2.2.2.1-5 토큰화된 문서의 리스트

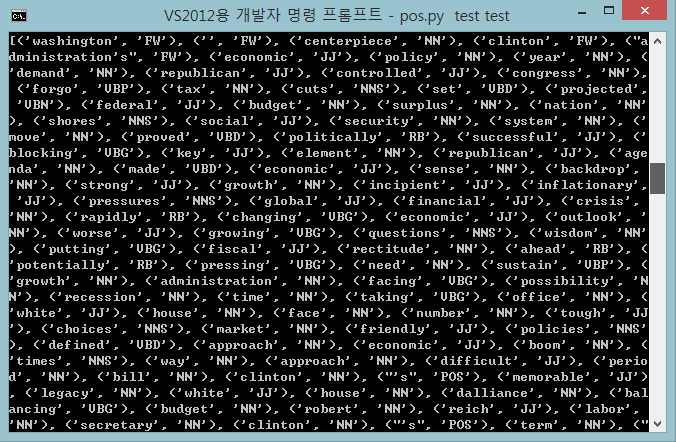


그림 2.2.2.1-6 각각의 토큰에 달린 품사태그

1. **쿼리 추출 함수**

토큰화 함수를 이용해 각각의 토큰에 tag를 달은 후에 쿼리를 추출하는 함수. 현재는 명사(NN, NNS)나 동사(VB)가 연속으로 3개가 있는 경우에 쿼리로 추출하는 방식을 사용하고 있으며 여러 실험들을 통해 효과적인 쿼리를 전송하는 방식을 찾고 있다.

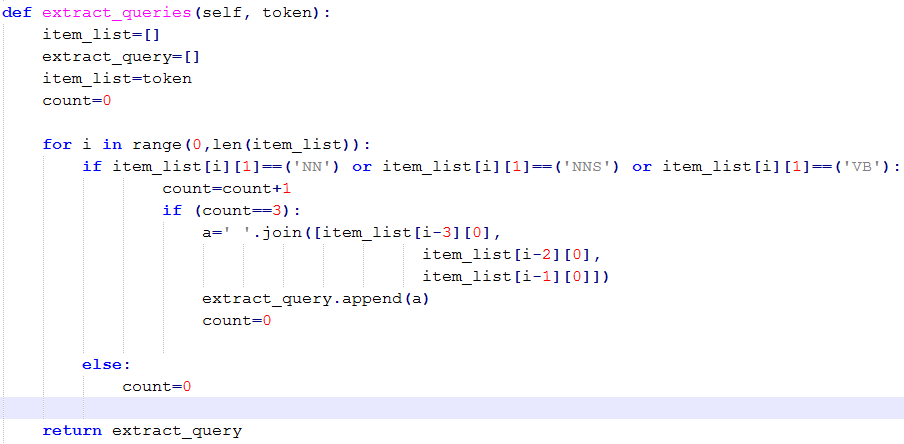


그림 2.2.2.1-7 쿼리 추출 함수

1. **로그 저장 함수**

쿼리를 보낸 시간, 쿼리, 검색 API에서 검색되는 URL을 기록하는 로그파일

총 2번 실행되며 첫 번째에 쿼리를 보낸 시간, 쿼리가 기록되며 두 번째에 검색 API에서 검색되는 URL을 기록한다.

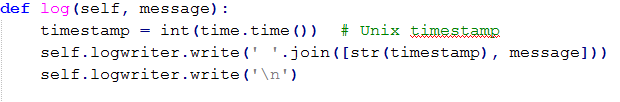


그림 2.2.2.1-8 보낸 쿼리를 로그로 저장하는 함수

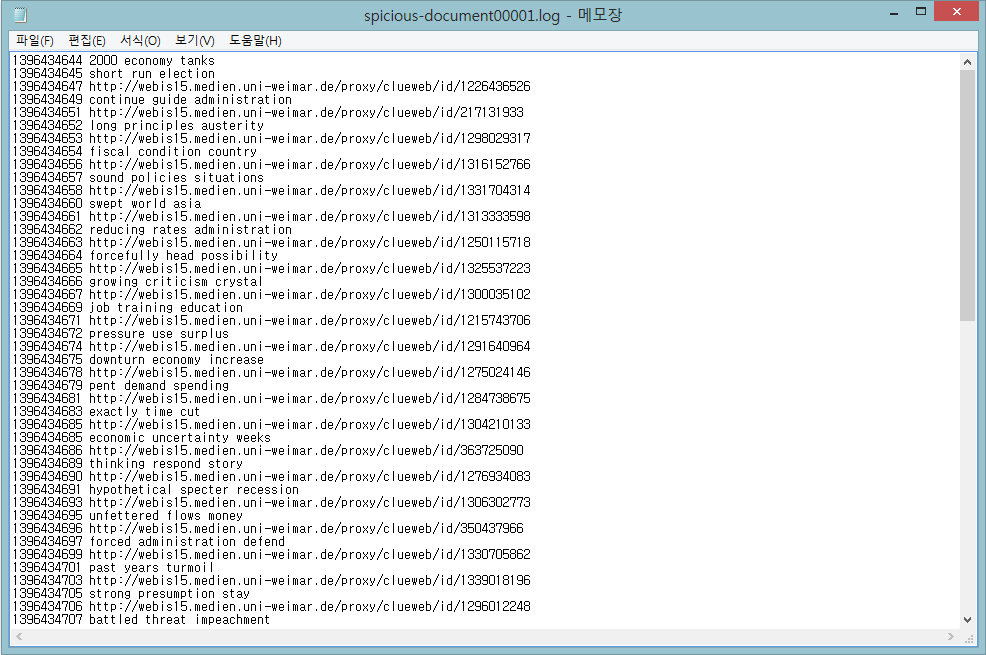


그림 2.2.2.1-9 로그파일

1. **결과 함수**

쿼리를 날린 후 검색 API에서 보내주는 결과.

결과가 없을 경우 다음 쿼리를 검색 API에 전송해주며 결과가 있을 경우 json형식의 결과를 리턴한다.

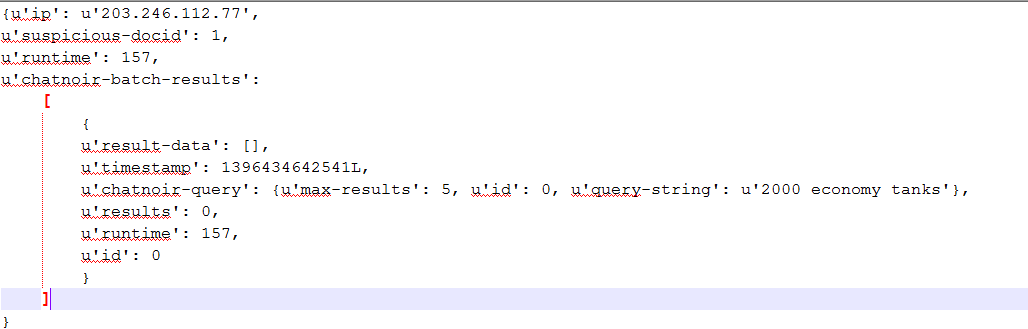


그림 2.2.2.1-10 전송한 쿼리에서 결과가 없을 경우



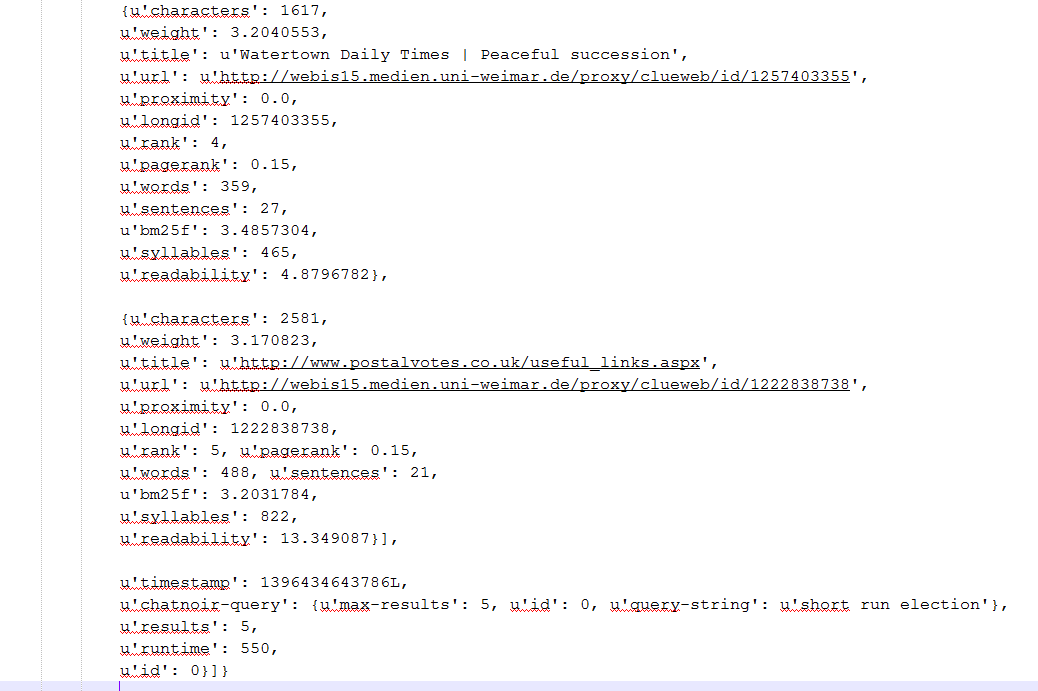


그림 2.2.2.1-11 전송한 쿼리에 결과가 있는 경우

1. **URL 다운 함수**

결과에서 가져온 Json형식의 파일을 이용하여 로그파일에 기록 후에 결과에 저장되어있는 첫 번째 URL을 다운로드 받아오는 함수. 차후에 첫 번째URL만 다운받는 방식이 아니라 좀더 효율적인 방식으로 바꿀 예정이다.

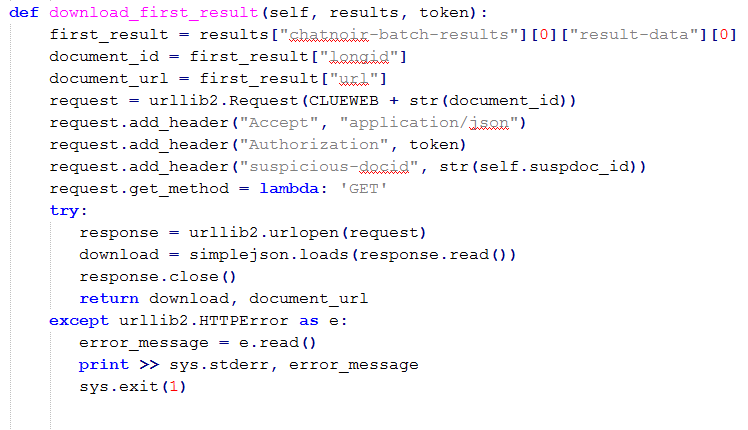


그림 2.2.2.1-12 URL 다운

1. **URL 비교 함수**

다운로드 받은 URL이 표절한 문서인지 서버API의 ORACLE을 이용하여 판별하는 함수.

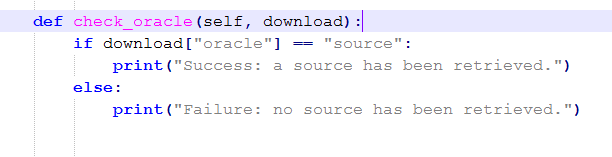


그림 2.2.2.1-13 URL 비교 함수

1. **TF-IDF 함수**

안정적인 Query추출을 위해서 사용하는 함수.

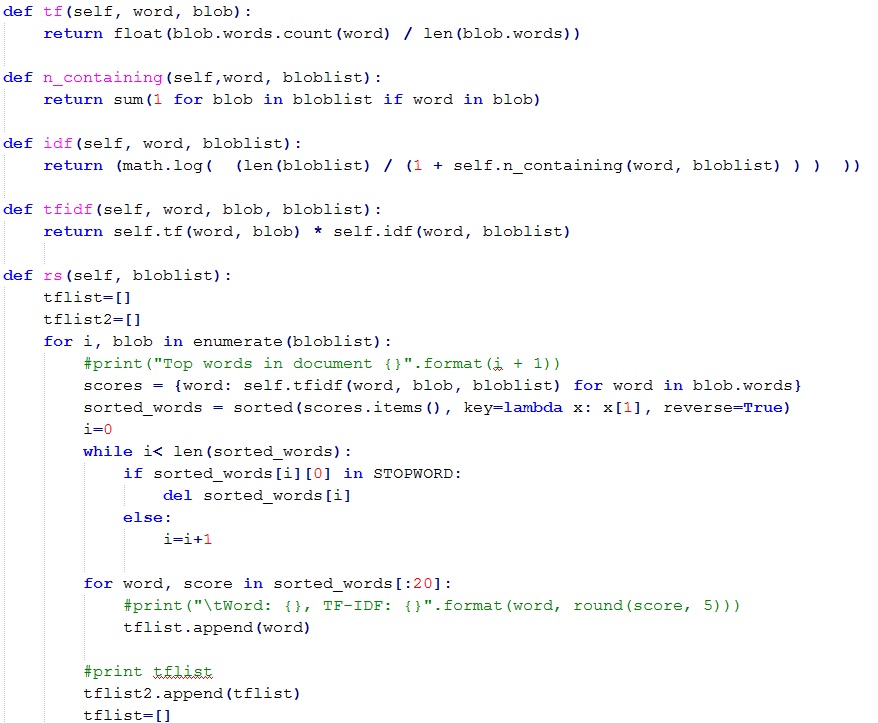


그림 2.2.2.1-14 TF-IDF 함수

1. **URL 다운 함수**

10번의 함수를 리스트화 하여 다운받을 수 있는 URL을 리스트화 시킴. 그 후 jsonquery를 통해서 다운로드 할 URL의 개수를 정해 줄 수 있다.



그림 2.2.2.1-15 그림2.2.1-12를 리스트화

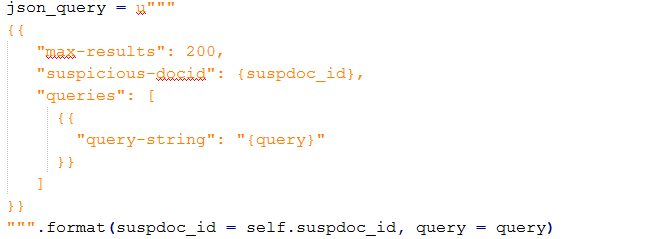


그림 2.2.2.1-16 다운로드 할 URL을 조절하는 json형식의 query

#### 2.2.2.2 Text Alignment

##### Text Alignment 개요

Source-Retrieval를 통하여 표절이 의심 문서가 표절했다고 생각되는 웹 상의 소스문서 후보군을 찾을 수 있다. 그리고 이 후보 문서들을 다운로드 받는다. Text Alignment 과정에서는 이 후보 문서들과 의심 문서를 정밀 검사한다. 의심 문서를 단어 별로 분석하고 소스 문서라고 생각되는 문서에서 어떤 부분을 표절했는가에 대한 결과를 출력한다. 표절이 의심되는 문서를 의심 문서, 의심 문서가 표절한 것으로 생각되는 문서를 소스 문서라고 칭하겠다.

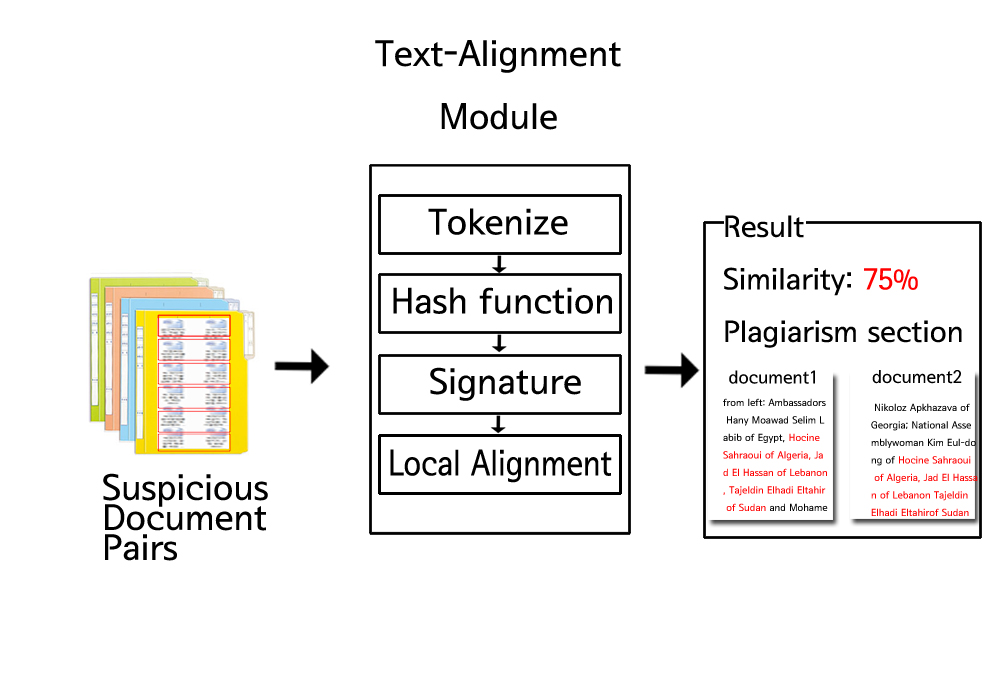


그림 2.2.2.2-1 Text-Alignment 1차 개념도

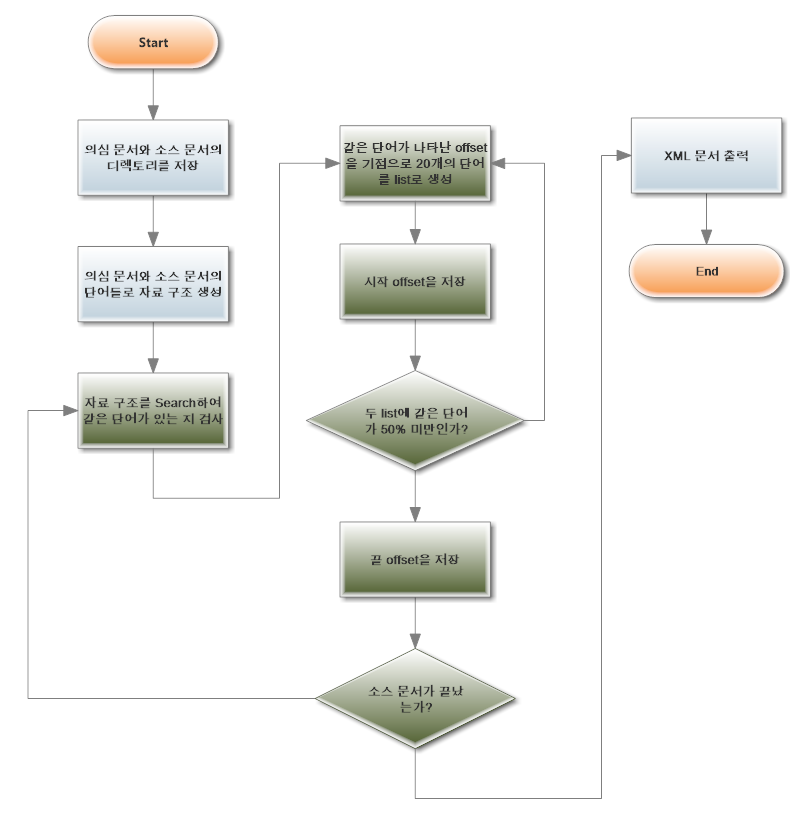


그림 2.2.2.2-2 Text Alignment 흐름도

##### Text Alignment 알고리즘

Text Alignment의 핵심 알고리즘은 다음과 같다.

1. **의심 문서와 소스 문서의 디렉토리를 저장한다.**
2. **의심 문서와 소스 문서에 대한 자료 구조를 생성한다.**
3. **각각의 자료구조를 비교해 동일 단어를 찾아낸다.**
4. **동일 단어부터 각각 20개의 단어를 추출하여 리스트를 만든다.**
5. **리스트를 각각 정렬하여 유사도를 비교한다.**
6. **일정 유사도를 초과하는 경우 이 구간이 표절이라고 가정한다.**
7. **표절 구간의 Offset을 저장한다.**

Text Alignment에서 동일 단어로부터 리스트를 추출하여 유사도를 비교하는 알고리즘은 다음과 같다. 다음은 그 과정을 나타낸 의사 코드이다.

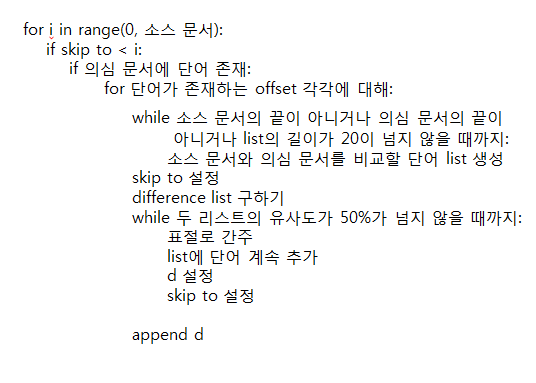


그림 2.2.2.2-3 Text Alignment 단어 리스트 추출 알고리즘

Text Alignment의 2차 개념도는 다음과 같다.

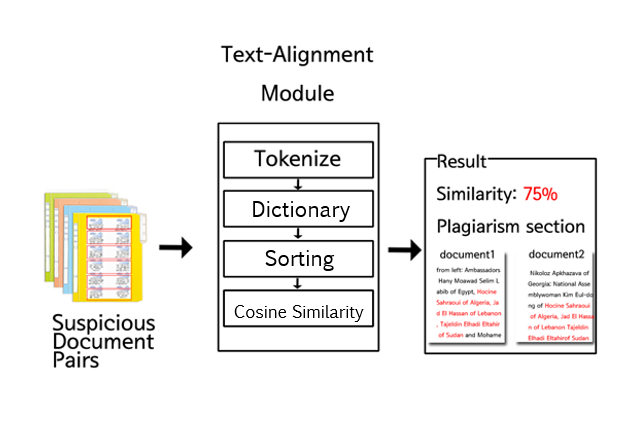


그림 2.2.2.2-4 Text Alignment 2차 개념도

preprocess, compare, postprocess의 세 가지로만 나뉘어 있던 모듈을 구체화 하였다. Preprocess의 과정은 소스 문서와 의심 문서를 단어 단위로 토큰화하는 함수로 나누었으며 compare은 구체적인 비교 부분으로써 세부 함수로 나눌 계획이다. Postprocess 또한 결과 XML 문서를 만드는 함수인 serialize\_features 함수로 구성되었다.

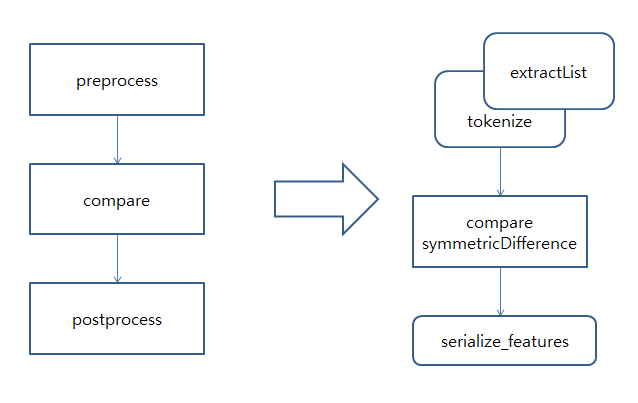


그림 2.2.2.2-5 Text Alignment 구체화된 모듈

문서 리스트의 유사도를 비교할 때 코사인 유사도 함수를 쓰지 않고 파이썬에 내장되어 있는 함수인 symmetric\_difference 함수를 사용하였다. 이로써 코사인 유사도 비교의 부정확성을 개선하였다.

기존의 유사도 검사는 cosine 유사도 함수를 이용하여 문서의 벡터 스페이스 모델을 추출하는 방식이었다. 하지만 이는 문서 간의 유사도가 어느 정도인지 사람이 직관적으로 이해하기 어렵고, 단어들의 배열에 따라 결과값이 다르게 나왔다. 이를 개선하기 위해 python의 symmetric\_difference 함수를 쓰도록 한다.

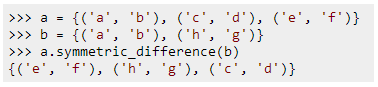


그림 2.2.2.2-5 symmetric\_difference 내장 함수

기존 cosine 유사도는 사람이 직관적으로 이해하기 어려웠지만, symmetric\_difference 함수를 씀으로써 두 리스트 간의 비교가 간편하게 이루어진다.

##### Text Alignment 알고리즘 구현

1. **사용 라이브러리**

NLTK (Natural Language Tool Kit) : NLTK 라이브러리 중 Stanford의 POS Tagger를 이용하였다. 이를 이용하여 단어의 품사를 분류하였으며, 형용사, 동사, 명사만을 중요 단어라고 간주하고 불용어를 제거하였다

1. **실제 구현 소스코드**

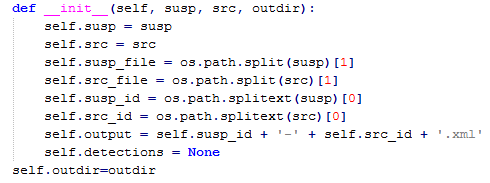


그림 2.2.2.2-6 Text Alignment 디렉토리 저장 함수

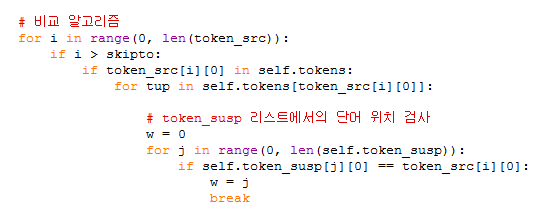


그림 2.2.2.2-7 두 문서에서 동일 단어를 추출하는 함수

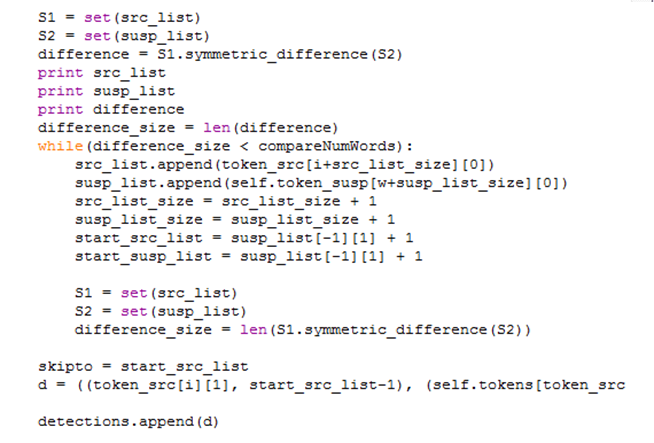


그림 2.2.2.2-8 Text Alignment 문서 유사도 추출 함수

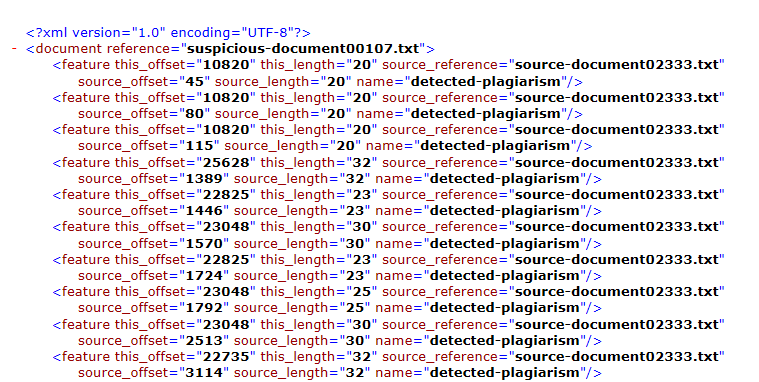


그림 2.2.2.2-9 Text Alignment 표절 구간 Offset 출력 XML

#### 2.2.2.3 User Interface

### 시스템 기능 및 구조 설계도

**개발한 SW/HW의 전체 구성에 대한 설명**

**계획서 보다 훨씬 자세해야 하며 설계 사양이 구체적이어야 함**

**도식, 시퀀스 다이어그램 등을 활용**

### 현실적 제한 요소 및 그 해결 방안

- 표절 문서 군의 부재

표절 성능 측정을 위한 많은 양의 표절 문서 군의 확보가 어렵기 때문에 성능측정을 하는데 어려움이 있었으니 PAN2014에서 제공하는 corpus를 이용하여 현재 비교할 수 있는 문서군을 확보하여서 성능측정에 관한 문제가 해결되었다.

- 라이센스

현재 사용하고 있는 라이브러리들의 상업적 사용은 불가하기 때문에 학술적인 목적으로만 이용할 수 있다. 향후 관련 사업을 진행하게 된다면 필히 확인해야 할 부분이다.

- 포맷의 다양성

문서 포맷인 doc, pdf, hwp, txt등 현재 사용되고 있는 문서들의 다양한 포맷들을 읽어 들일 수 있는 필터가 필요한데 직접 돈을 주고 사기에는 학생입장에서는 워낙 큰 금액이기 때문에 일단 만들어 보기로 합의를 하였다.

- Web Search API의 제공 기간

현재 쓰고 있는 Web Search API경우 일시적으로 사용기간을 부여하기 때문에 대체할 수 있는 Web Search API를 찾아야 한다.

- 표절 검사구간의 정확성

글을 표현하는 주체는 사람으로 사람마다 표현하는 방식이 다 달라 거기서 70%이상의 정확성을 찾기 힘들다. 그리고 같은 문장을 다른 순서만 배치를 다르게 하는 경우, 요약을 해서 표현을 하는 경우, 뜻은 같지만 단어 형태가 다른 경우(동음이의어)에 대한 구간을 찾는 것은 힘들다. 이런 구간을 찾으려고 할 시 많은 예외 처리가 필요하며 예외처리를 하려고 할 시 이에 대한 데이터가 Database에 저장이 되어 있어야 한다. 많은 데이터를 저장함으로써 오차를 줄이고 정확성을 높일 수 있다.

### 결과물 목록

**프로젝트 수행의 결과물을 목록으로 제시하고 이에 따른 기술문서 유/무를 표시 한다.**

**기술 문서가 있는 경우 부록에 삽입한다.**

# 자기평가

## 기대효과 및 활용방안

최근 많은 표절 및 유사 행위로 인하여 사회적 혹은 도덕적으로 문제가 야기되고 있다. 사회 전반적으로 표절에 대한 경각심은 크게 찾아볼 수가 없으며 오히려 만연해 있는 상황이다. 표절에 의한 많은 피해 사례나 국외에서도 표절된 논문 적발과 같은 많은 문제가 발생하고 있으며, 표절검사가 제대로 이루어지지 않음에 따라 창조적인 연구가 진행되지 않고, 간단하게는 학생들의 숙제부터 심각하게는 논문에 이르기까지 표절이 이루어지고 있다는 점은 장기적인 관점에서 매우 심각한 문제이며 꼭 해결해야만 하는 문제라고 볼 수 있다. 이러한 상황에서 표절 예방 및 표절 여부를 판단하는 것 역시 중요한 사항으로 떠오르게 되었다.

본 프로젝트를 통하여 개발된 COPTDET 시스템은 주어진 문서에서 키워드를 추출한 후 유사한, 표절의 의심되는 문서들을 확인하여 표절된 부분이 있는지 확인하고 표절이 확인되면 표절된 부분을 출력해준다. 이 표절검사 프로그램을 사용함으로써 온갖 문서들의 표절을 누구나 손쉽게 찾아 낼 수 있다. 이를 통해 사회 전반적으로 만연해있는 표절에 대한 관대함과 표절을 당연시 여기는 사회풍토를 지양할 수 있고 표절에 대한 경각심을 키울 수 있다. 그리고 표절검사를 시스템화 함으로써 사용되는 인적자원 및 시간들을 획기적으로 줄일 수 있다.

표절 논란을 종식시키는 길잡이로서 COPYDET은 교육계를 넘어 사회 전반에 널리 기여할 수 있다. 향후 각 대학, 연구기관, 공공기관, 신문, 방송, 잡지사 등 사회 각 분야에 공급하여 올바른 저술 활동을 장려하고 창의적인 연구 및 저술 활동을 지원하는데 활용 되어 알맞은 인용과 참조 문화가 사회에 자리 잡을 수 있도록 기여한다.

## 자기평가

### 유사서비스 현황

현재 국내에서 서비스되고 있는 표절검사 서비스를 보면 대표적으로 카피킬러(CopyKiller), 밈체커(MemeChecker), 카피레스(COPYLESS)가 있다. 카피킬러와 밈체커의 표절검사 서비스는 웹 기반으로 서비스 중이다. 카피킬러의 경우 21개의 대학과 제휴하여 각 대학별 맞춤 서비스를 통하여 서비스를 운영 중이고, 밈체커의 경우 5개 이상의 대학과 기타 연구기관과 제휴하여 서비스를 운영 중이다. 웹 상에 사용자가 작성한 문서를 업로드 하면 각 서비스 사업자의 DB와 인터넷상의 웹 문서에서 표절 여부를 검사한 후 결과를 보여주는 식으로 운영되고 있다. 카피레스의 경우는 윈도우 응용프로그램의 형태로 원하는 문서를 선택하면 DAUM과 NAVER의 API를 이용 관련 의심문서를 Microsoft Access를 이용하여 DB화하고 문서화 DB를 확인하여 결과를 보여주는 형태로 운영 중이다.

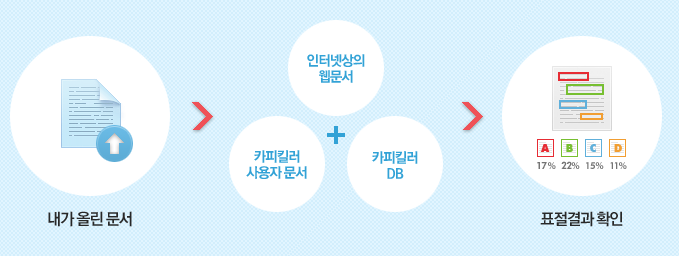


그림 3.2.1-1 유사서비스의 서비스 모델

### 사용가능성

본 프로젝트에서 개발된 COPYDET 시스템의 경우 핵심 모듈인 Source Retrieval과 Text Alignment 모듈의 Recall과 Precision을 최적화하는 방법에 있어서 Heuristic한 방법으로 진행이 되면서 프로젝트 진행이 다소 늦어진 측면으로 인해 UI구현과 각 모듈의 연동과정에서 다소 완성도가 부족하게 진행되었지만 실험을 통한 모듈의 최적화로 향후 서비스를 운영 시 경쟁력을 확보 할 수 있을 것으로 기대된다..

### 향후 과제

Source Retrieval과 Text Alignment 모듈의 경우 현재 Heuristic한 방법으로 최적화하였으므로 성능 개선의 여지가 남아있으므로 지속적인 실험과 새로운 아이디어의 융합을 통한 성능 향상을 목표로 지속적으로 개선해 나갈 것이다.

향후 가장 시급한 과제로 UI의 사용성 개선이 필요하다. 현재 구현된 모듈의 핵심 기능을 단순히 확인하기 위한 형태로 UI를 구현하였다. 향후 COPYLESS 시스템과 같이 윈도우 응용프로그램의 형태 혹은 카피킬러, 밈체커의 형태로 웹사이트 상에서 서비스를 구현 할 것인지 향후 서비스 모델이 특정되는 것에 따라 추진할 계획이다.

또한 서비스 운영적인 측면에서 각 대학 및 교육기관의 이러닝시스템, 도서관의 학술 정보시스템, 연구기관의 연구자료 평가시스템, 기업의 채용관리시스템, 입시기관의 입시평가가 시스템 등에서 COPYDET시스템과 연동하여 사용할 수 있도록 진행하고 있다.

# 참고 문헌

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 번호 | 종류 | 제목 | 출처 | 발행년도 | 저자 | 기타 |
| 1 | Web | 구글 논문검색 (http://scholar.google.co.kr/) |  |  |  |  |
| 2 | Web | 네이버 논문검색  (http://academic.naver.com/) |  |  |  |  |
| 3 | Web | 학술연구정보서비스  (http://www.riss.kr/) |  |  |  |  |
| 4 | 서적 | 두산백과 |  | 2013 |  |  |
| 5 | 서적 | 열혈강의 파이썬 | 프리렉 | 2005 | 이강성 |  |
| 6 | 서적 | Head First Python | 한빛미디어 | 2011 | Paul Barry |  |
| 7 | 논문 | 대학생의 글쓰기 과제물의 표절 실태와 표절 검사 시스템의 표절 예방 및 적발 효과 연구 |  | 2010 | 정소연 외 6명 |  |
| 8 | 논문 | 저작권 및 저작권 교육에 관한 초등학생의 인식조사 | 경남대학교 교육대학원 | 2011 | 조희경 |  |
| 9 | 논문 | Plagiarism Candidate Retrieval Using Selective Query Formulation and Discriminative Query Scoring | Nile University, Cairo, Egypt | 2013 | Osama Haggag 외 1명 |  |
| 10 | 논문 | CopyCaptor : Plagiarized Source Retrieval System using Global word frequency and Local feedback | Korea University | 2013 | 이태민 외 3명 |  |
| 11 | 논문 | Approaches for Source Retrieval and Text Alignment of Plagiarism Detection | Heilongjiang Institute of Technology, China | 2013 | Kong Leilei외 4명 |  |
| 12 | 논문 | External and Intrinsic Plagiarism Detection using Vector Space Models | Graz University of Technology | 2009 | Mario Zechner외 1명 |  |
| 13 | 논문 | Text Alignment Module in CoReMo 2.1 Plagiarism Detector | Universidad de Huelva | 2013 | Diego A. Rodríguez Torrejón 외 1명 |  |
| 14 | 논문 | 오픈 소스코드 표절 탐지 기법 | 이화여자대학교 | 2009 | 한소정 |  |
| 15 | 논문 | 문서와 프로그래밍언어의 표절검사 기술에 관한 연구 | 유한대학 | 2007 | 김영철 |  |
| 16 | 논문 | 한국어 문서 표절 검사를 위한 LSA와 N-gram 기반의 유사 문장 판별 방법 | 고려대학교 | 2010 | 지혜정, 조준희, 임희석 |  |
| 17 | 논문 | 인터넷 검색과 형태소분석을 이용한 표절검사시스템의 개발에 관한 연구 | 전주대학교 | 2009 | 황인수 |  |
| 18 | 논문 | 문서구조 정보 기반의 유사도 측정 | 공주대학교 | 2010 | 신미해 외 3명 |  |
| 19 | 논문 | 문서와 프로그래밍 언어의 표절 검사 기술에 관한 연구 | 유한대학 | 2007 | 김영철 |  |
| 20 | 논문 | 문서 구조 정보를 이용한 유사도 검사 시스템의 설계 및 구현 | 숭실대학교 | 2008 | 박주열 |  |
| 21 | 논문 | 구문분석기법을 이용한 문서표절탐지 알고리즘의 개발 | 안동대학교 | 2013 | 박상욱 외 5명 |  |
| 22 | 논문 | 대용량 문서 집합에서 유사 문서 탐색을 위한 효과적인 전처리 시스템의 설계 | 부산대학교 | 2010 | 박선영 외 4명 |  |
| 23 | 논문 | DNA 염기서열의 유사성 검색을 위한 효율적인 알고리즘 | 전남대학교 | 2011 | 김병기 외 4명 |  |
| 24 | 논문 | 실시간 리포트 표절 검사를 위한 임계 알고리즘 개발 연구 | 경희대학교 | 2010 | 임나리 |  |

# 부록

## 사용자 매뉴얼

**설치 가이드, 따라하기 등 포함**