한글 말뭉치를 이용한 한글 표절 탐색 모델 개발

(Developing of Text Plagiarism Detection Model using Korean Corpus Data)

류 창 건 † 김 형 준 †
(Ryu Chang-keon) (Kim Hyong-Jun)

조 환 규 ††
(Cho Hwan-Gue)

요 약 최근 들어 각종 창작물에 대한 표절 사건이 빈번하게 발생하고 있다. 특히 문서들 간의 표절은 현재 많은 이슈가 되고 있다. 영어에 관한 표절연구는 서양에서 오래 전부터 이뤄져 왔지만 한글은 구조적인 어려움으로 인해 아직 많은 연구가 이뤄지지 않고 있다. 한글은 영어와 구조적인 특징이 많이 다르기 때문에 영어기반의 탐색 기법을 한글 문서에 적용하기는 어렵다. 본 논문에서는 한글의 특성에 맞는 새로운 표절 탐색 기법을 소개하고 한글 말뭉치를 이용하여 그 성능을 실험해본다. 제안된 기법은 "k-mer"와 "지역정렬" 방법을 기반으로, 문서들 간의 표절구간을 매우 빠르고 정확하게 찾아낸다. 또한 우리는 천만어절 이상의 크기를 가진 한글 말뭉치를 이용하여 표절이 일어나지 않은 일반적인 문서에서 우연히 나타나게 될 유사 확률에 관한 모형을 만들었다. 시스템을 이용하여 성능을 측정해 본 결과, 표절 문서를 매우 정확하게 찾는 것을 알 수 있었다.

키워드 : 표절 탐색, 한글 말뭉치, 정보 검색

Abstract Recently we witnessed a few scandals on plagiarism among academic paper and novels. Plagiarism on documents is getting worse more frequently. Although plagiarism on English had been studied so long time, we hardly find the systematic and complete studies on plagiarisms in Korean documents. Since the linguistic features of Korean are quite different from those of English, we cannot apply the English-based method to Korean documents directly. In this paper, we propose a new plagiarism detecting method for Korean, and we throughly tested our algorithm with one benchmark Korean text corpus. The proposed method is based on "k-mer" and "local alignment" which locates the region of plagiarized document pairs fast and accurately. Using a Korean corpus which contains more than 10 million words, we establish a probability model for local alignment score (random similarity by chance). The experiment has shown that our system was quite successful to detect the plagiarized documents.

Key words : Plagiarism detecting, Korean Text corpus, Information retrieval

1. 연구 동기

최근 들어 표절과 관련된 많은 사건들이 발생하여 사회적 문제가 되고 있다. 이에 문서들 간의 표절을 자동으로 찾아주는 시스템의 필요성이 부각되고 있다. 영어 문서의 표절에 대한 연구는 오래전부터 시작됐지만 한글은 구조적인 특성으로 인해 아직 많은 연구가 이뤄지지 않고 있다. 실제 현재 한글의 특성에 맞게 개발된 표절 탐색 시스템은 널리 알려진 것이 없어 외국의 유명 표절 방지 시스템을 사용하는 경우가 많다[1]. 하지만 이 시스템들은 주로 영어를 대상으로 개발되었기 때문에 한글 문서의 표절 탐색에는 큰 효과를 거두지 못하고 있다.

영어와 한글의 가장 큰 차이점은 말의 어순이 다르다는 것이다. 영어는 정해진 어순을 다르게 쓸 수 없지만 한글은 어떤 순서로 사용하여도 그 의미가 통하며 단지 강조하는 부분이 달라지는 효과를 낸다. 예를 들어 한글은 "철수는 학교에 간다."라는 문장을 "학교를 철수가 간다.", "철수가 가는 곳은 학교이다."와 같이 어순을 바꿔도 동일한 뜻을 갖지만 영어에서는 "Chul-su goes to the school."이라는 문장을 "School goes to the Chul-su."로 바꾸게 되면 다른 의미를 가지게 되므로 어순을 바꿀 수 없다. 즉, 영문서와 달리 한글 문서의 표절 탐색을 위해서는 어순이 다양하게 바뀌어도 표절을 찾아낼 수 있는 시스템이 필요하다.

그리고 영어는 단어 단위로 띄어쓰기가 되어 있어 문서의 비교를 위해 띄어쓰기 단위로 나누면 되지만, 한글은 띄어쓰기가 어절별로 되어 있어 문서의 비교를 위해서는 어절을 형태소 분석을 통해 어간과 어미로 나누는

· 본 논문은 부산대학교 교내 학술연구비(4년 과제)에 의한 연구임
· 이 논문은 제34회 추계학술대회에서 '한글 말뭉치를 이용한 한글 표절 탐색 모델 개발'의 제목으로 발표된 논문을 확장한 것임

† 학생회원 : 부산대학교 컴퓨터공학과
ckryu@pearl.cs.pusan.ac.kr
hjkim83@pearl.cs.pusan.ac.kr
†† 정 회 원 : 부산대학교 컴퓨터공학과 교수
hgcho@pusan.ac.kr

논문접수 : 2007년 12월 7일
심사완료 2008년 2월 14일

정보과학회논문지: 컴퓨팅의 실제 및 레터 제14권 제2호(2008.4)

전처리 과정이 필요하다. 마지막으로 영어와 한글은 컴퓨터에 저장되는 크기가 달라 영어를 비교하기 위해 *1byte* 비교방식으로 만든 표절 탐색 시스템은 *2byte*로 저장되는 한글에서는 사용할 수 없다. 물론 유니코드로 비교하는 표절 탐색 시스템은 예외이다. 이와 같은 영어와 한글의 다양한 차이점으로 인해 한글에 맞는 표절 탐색 시스템의 개발이 필요하다.

이에 본 논문은 한글 문서에 맞는 표절 탐색 시스템을 제안하고, 한글 말뭉치를 이용하여 한글의 표절 확률 모델을 제안하고자 한다. 또한, 말뭉치를 이용한 실험을 통해 문서간의 유사도를 정규화된 확률 모델과 비교하여 표절 여부를 판별하고자 한다. 그리고 웹문서와 리포트 등 여러 분야의 문서들을 대상으로 한 실험을 통해 표절 탐색 시스템의 성능과 이용 가능성을 보이고자 한다.

2. 관련 연구

한글 문서의 표절 탐색 방법은 아직 많은 연구가 이뤄지지 않았으나, 영어 문서에 대한 표절 탐색 방법은 이전부터 많은 연구가 있었고, 크게 두 종류로 구분할 수 있다.

첫 번째 방법은 Attribute counting방법으로, 문서에서 자주 사용되는 단어들 간의 유사성이나 빈도수를 측정하여 표절 정도를 나타내는 방법이다[2]. 표절 탐색 시간이 문서의 길이에 영향을 받지 않으며 문서들 간의 표절 검사에 유용하게 사용된다. 또한 문서의 단락 순서들을 섞어도 표절 탐색 성능에 영향을 받지 않는 장점을 가진다. 하지만 부분 표절 탐지가 어렵다는 단점을 가지고 있다. 대표적인 방법에는 CloneChecker가 있으며 이 방법은 표절을 탐색할 때 문서 요약 단계와 문서 간 유사성 비교 단계, 유사한 것끼리 그룹 짓는 단계로 나눠서 표절을 탐색한다[3].

두 번째 방법은 Structure metric방법으로 측정되는 단어의 정확한 일치(match)가 아닌 토큰 스트링(Token string)의 유사성을 계산하여 표절 탐색을 하는 방식이다. 문서의 길이와 얻어내는 정보가 비례하기 때문에 문서의 길이는 표절 탐색 시간에 영향을 주며, 문서의 단락의 순서를 변경하면 표절 탐색 성능이 나빠진다. 부분 표절 탐지가 용이하며, 프로그램 소스 코드를 검사할 때 효과적으로 사용된다. Attribute counting 방법보다 더 효율적인 방법으로 평가되고 있으며 현재 공개된 표절 탐색 시스템의 대다수가 이 방식을 채택하고 있다. 대표적인 방법으로는 'Plague', 'SIM', 'YAP' 등이 유명하다. 'Plague' 방법은 유동성이 부족하여 새로운 언어를 대상으로 표절 탐색을 하기 어렵다[4]. 'SIM' 방법은 기본 단위를 토큰으로 하며 연속되는 여러 개의 토큰으로 구성되는 런(run)을 비교 단위로 하여 두 파일을 비교한다[5]. 'YAP' 방법은 'YAP3' 방법으로 향상되었으며 스트링 매칭 방법을 사용하여 소스 문서를 토큰 시퀀스로 변경시키는 단계와 토큰 스트링을 서로 비교하는 단계를 가진다[6]. 국내에서는 고려대학교, 한양대학교에서 표절탐색에 관한 연구가 진행되고 있으며 대학내부에서 활용되고 있다고 한다.

3. 한글 표절 탐색 시스템의 구조

본 논문에서 제시하는 한글 표절 탐색 시스템은 'DEVAC(Document EVolution Analyzing Center)' 시스템이라 불리며, 한글의 특성에 맞게 시스템이 동작할 수 있도록 'fingerprint' 방식과 'BLAST' 방식을 혼합하여 사용하고 있다[7]. 'fingerprint' 방식은 문서의 인덱스를 두어 빠르게 탐색하는 방식이며 'BLAST' 방식은 대용량의 데이타에서 유사한 부분을 정확하고 빠르게 찾는 방식이다[8,9]. DEVAC 시스템은 이 두 방법을 응용하여 대용량 한글 문서에서도 빠르고 정확하게 표절 영역을 찾아낸다. 그림 1은 DEVAC 시스템의 전체 흐름을 나타낸 흐름도이며 진행 단계는 다음과 같이 3단계로 나누어진다.

첫 번째는 전처리 단계로써 문서간의 빠른 비교를 위해 입력 문서를 정해진 형식으로 변환하는 과정이다. 표절 탐색할 문서들은 동일한 파일형식으로 저장되어 있지 않을 경우가 많으므로, hwp, html, doc, pdf 등의 파일에서 텍스트만 추출하여 DEVAC 시스템에서 정의한 표준 포맷 형식인 stx파일로 변환한다. stx파일은 문서에 라인번호를 부여하여 각 어절의 위치를 정하며, 문서

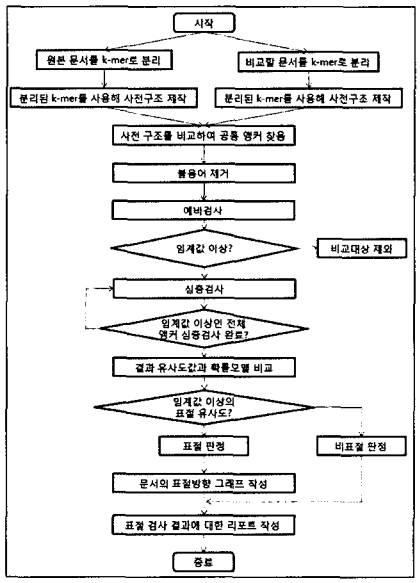

그림 1 한글 표절 탐색 시스템의 전체 흐름도

표 1 k-mer 단위로 나눠진 사전 구조

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
새	야		새	야		파	랑	새	야		녹	두	발	에		앉	지	마	라

2-mer dictionary				
녹두	12			
두발	13			
랑새	8			
마라	19			
발에	14			
새야	1	4	9	
앉지	17			
지마	18			
파랑	7			

3-mer dictionary				
녹두발	12			
두발에	13			
랑새야	8			
새야	1	4	9	
앉지마	17			
지마라	18			
파랑새	7			

4-mer dictionary			
녹두발에	12		
새야	1	4	
앉지마라	17		
파랑새야	7		

의 크기, 라인수, 글자수 등 여러 가지 정보를 저장하고 있다. 표준 파일로 변환된 이 파일을 다시 k-mer 단위로 나누어, 표절을 탐색하기 쉬운 사전 구조로 만들게 된다[10]. 표 1은 k-mer 단위로 나눠진 사전 구조를 나타낸다. 각 k-mer에 따라 구성되는 사전은 다른 모형을 갖게 되며 이는 표절 탐색 성능에도 영향을 주게 된다.

두 번째는 표절을 탐색하는 단계로써 전처리 단계에서 생성된 표준 파일과 사전 구조를 이용하여 빠르게 표절 영역을 찾는 과정이다. 원본 문서와 비교 문서의 사전 구조를 비교하여 공통 앵커를 찾은 후, 표절 검사를 하지 않아도 되는 불용어를 찾아 제거하게 된다. 불용어에는 문서의 흐름에 큰 영향을 주지 않으나 빈번하게 반복해서 나타나는 조사와 동일한 주제로 작성되거나 동일한 책을 인용하여 글들이 작성되었을 경우에 나타나는 반복 주제어가 있다. 불용어를 제거한 후, 실제 표절 탐색 단계를 실시하게 되는데 이 단계는 크게 두 단계로 나뉜다.

첫 번째 단계는 예비검사 단계로 앵커를 기준으로 문서의 표본 영역만을 추출하여 빠르게 유사도를 측정한다. 이 단계에서 표본 영역의 유사도가 작은 앵커들을 비교 대상에서 제거시켜 다음 수행 단계의 부하를 줄인다. 두 번째 단계는 심층검사 단계로 예비검사를 통과한 앵커들에 대해 지역정렬 방식을 사용하여 보다 정밀한 표절 영역 탐색을 하게 된다. 표 2는 두 문장의 지역정렬 과정을 나타낸 표이다. 지역정렬 방식은 두 어절이 일치하면 가점을 주고 두 어절이 다르면 감점을 주는 방식으로 연속적으로 유사한 어절이 많은 영역을 쉽게 찾을 수 있다.

표 2 두 문장의 지역정렬 과정

		당신의	고요한	가슴으로	닥쳐오는	슬	가른	벗들의	말발굽	소리	누가	내게
	0	0	0	0	0	0	0	0	0	0	0	0
당신의	0	8.13	3.63	0	0	0	0	0	0	0	0	0
고요한	0	2.63	16.26	11.76	7.26	2.76	0	0	0	0	0	0
가슴으로	0	0	10.76	26.57	22.07	17.57	13.07	8.57	4.07	0	0	0
닥쳐오는	0	0	5.26	21.07	36.89	32.39	27.89	23.39	18.89	14.39	9.89	5.39
슬	0	0	0	15.57	31.39	39.89	35.39	30.89	26.39	21.89	17.39	12.89
가른	0	0	0	10.07	25.89	34.39	45.59	41.09	36.59	32.09	27.59	23.09
자연의	0	0	0	4.57	20.39	28.89	40.09	43.59	39.09	34.59	30.09	25.59
생명의	0	0	0	0	14.89	23.39	34.59	38.09	41.59	37.09	32.59	28.09
소리	0	0	0	0	9.39	17.89	29.09	32.59	36.09	47.29	42.79	38.29
누가	0	0	0	0	3.89	12.39	23.59	27.09	30.59	41.79	52.99	48.49
내게	0	0	0	0	0	6.89	18.09	21.59	25.09	36.29	47.49	58.69

지역정렬을 이용한 심층검사 단계를 거치면 문서 내의 모든 표절 의심 부분들을 찾을 수 있다.

세 번째는 표절 의심 영역들을 한글 말뭉치를 통해 구성된 확률 모델과 비교하여 표절을 판정하고 결과 리포트를 작성하는 단계이다. 우리는 대량의 한글 말뭉치를 정제 작업을 거쳐 순수 비표절 문서로 만든 후, 이 문서들 사이의 유사도를 측정하여 확률 모델을 만들었다. 이 모델은 비표절 문서의 특징을 가지고 있어 표절 시스템의 결과에서 표절을 찾는 기준으로 사용된다.

4. 한글 말뭉치를 활용한 실험

한글 표절 탐색 시스템의 개발과 성능 측정을 위해서는 표절 영역이 전혀 없는 정규화 문서가 필요하다. 이를 위해 국립국어원에서 구축한 대량의 한글 말뭉치를 모두 수작업으로 검증하여, 표절이 완전히 제거된 비표절 독립(plagiarism-free independent) 문서 집합을 테스트용 자료로 재구성하였다.[11]. 이 문서의 일반적 특성을 알아내어, 표절이 일어난 문서와 차이점을 찾는다면 표절 문서와 비표절 문서의 구별을 쉽게 할 수 있다.

4.1 말뭉치의 유사도 확률 분포

정제 과정을 거친 말뭉치를 2000어절 단위로 200개의 문서들로 나눈 후, 전체 쌍에 대해 유사도 값의 분포를 그래프로 그려보니 그림 2와 같이 나타났다.

실험을 통해 이 분포가 포아송(Poisson) 그래프와 유사하다는 것을 알 수 있었다. 포아송 그래프는 평균값과 최대 꼭짓점 값이 일치하는 특성을 가지며, 기존에 계산된 표를 통해 각 유사도 값이 나타날 확률을 쉽게 알 수 있다. 만약 그림 2의 그래프가 포아송 그래프와 동일하다는 것을 증명할 수 있다면, DEVAC 시스템은 이 그래프를 표절 탐색 결과와 비교하여 문서의 표절 여부를 확률 값으로 쉽게 표시할 수 있게 된다. 현재 말뭉치를 이용하여 일반적인 한글 문서가 포아송 그래프의 확률 모델과 일치하는지 다양한 실험을 통해 검증하고 있다.

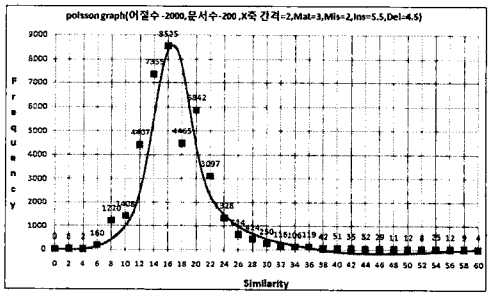

그림 2 한글 말뭉치의 절대유사도 분포도

기존 표절 탐색 시스템은 단순히 동일한 단어의 개수를 세어 두 문서의 유사도를 나타내었으나 DEVAC 시스템은 확률 모델을 이용하여 보다 객관적인 유사도를 제시하려 한다.

4.2 표절 탐색 시스템의 Specificity와 Sensitivity

표절 탐색 시스템의 성능을 실험하기 위해 우리는 115개의 표절 데이타 집합을 제작하였다. 각 집합은 원본 문서와 이를 순차적으로 표절한 5개의 문서로 이루어져 있다.

표절 유사도를 그래프로 나타낼 경우, 각 문서는 vertex가 되며 각 표절 데이타 집합마다 6개의 vertex가 생성된다. 6개의 vertex에서 가장 유사한 한 문서에 edge를 연결할 경우, degree는 1이 되며, 두 번째 유사한 문서와 edge를 연결하면 degree가 2가 된다. 만약 6개의 문서들의 가장 유사한 값들을 비교하여 가장 유사도가 큰 한 쌍의 문서에 edge를 연결하게 되면 degree는 6을 나눈 1/6 = 0.17이 된다. 6개의 표절 문서로 이루어진 115개의 표절 데이타 집합을 degree를 0.17에서 5까지 변경하면서 Specificity와 Sensitivity를 조사하여 그림 3과 같은 그래프를 얻었다. 여기서 Specificity와 Sensitivity는 아래의 수식으로 계산된다.

$$Specificity = \frac{\text{찾은것 중 표절쌍이 맞는 경우}}{\text{찾은 표절 문서쌍}}$$

$$Sensitivity = \frac{\text{조사결과 표절쌍을 찾은 수}}{\text{표절이 일어난 문서쌍}}$$

Specificity는 표절 탐색 시스템이 표절로 판정한 문서들이 실제로 표절 문서들이 맞는지를 측정하는 값으로, 표절 탐색 시스템의 명확성을 알 수 있다.

Sensitivity는 실제로 표절이 이루어진 문서 쌍들을 얼마만큼 표절 탐색 시스템이 잘 찾아내는지를 측정하는 값으로서, 표절 탐색 시스템의 민감성을 알 수 있다.

그림 3을 살펴보면 Sensitivity는 degree가 높아질수록 급격히 증가하여 degree가 2가 될 때, 95%에 도달하는 것을 알 수 있다. Specificity는 지속적으로 감소하는 곡선을 보이고 있으며 두 곡선이 만나는 1.33 값이 최적 값이다. 만일 사용자가 표절 문서를 놓치고 싶지 않다면 degree를 좀 더 올려 Sensitivity를 높이면 될 것이며, 시스템의 정확성을 올려서 비표절자가 표절자로 인식되는 경우를 방지하고 싶으면 degree를 낮추어 Specificity를 높이면 될 것이다.

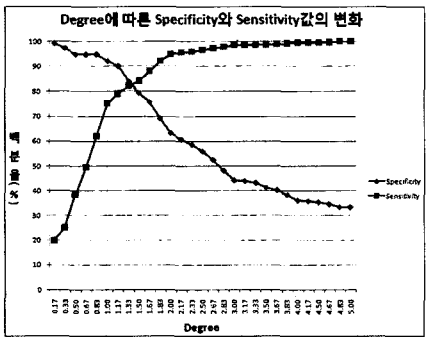

그림 3 표절 문서 사이의 degree에 따른 Specificity와 Sensitivity값 변화

5. 표절 시스템을 적용한 예

한글 표절 탐색 시스템은 사회 여러 분야에 응용하여 사용이 가능하다. 포털 사이트에 언론사들이 무분별하게 전송하는 기사들의 유사도를 조사하여 필터링하는 기능을 제공할 수 있으며, 학교에서 학생들의 리포트의 표절 방지에 사용할 수 있다. 또한 다른 논문을 표절하거나 짜깁기하는 행위를 표절 탐색 시스템을 통해 규제하여 이 같은 행위를 줄일 수 있을 것이다.

5.1 인터넷 검색 결과 군집화(clustering)

무분별하게 전송되는 중복 기사들을 여과 없이 보여주는 국내 포털 사이트에 비해 구글 뉴스는 중복 기사를 필터링하여 제공하고 있다. 하지만 실험한 결과 구글의 필터링도 완벽하게 중복을 제거하지는 못하였다.

표 3은 구글 뉴스에서 '장미희'라는 키워드로 검색된 10개의 문서들의 출처이며 그림 4는 표 3의 문서들을 표절 탐색한 결과를 나타낸 그래프이다.

그림 4에서 edge의 숫자는 두 문서 사이의 거리를 나타내며 0에서 1사이의 값을 갖는다. 거리가 가까울수록 표절이 많이 발생한 것으로 0값은 전체가 동일한 문서이다. 동일한 사건을 대상으로 기사를 작성하였으므로, 부분 표절된 문서는 제외하고 전체를 표절한 문서들끼리 군집으로 묶었다. 문서 집합 S1={장미희3, 장미희8, 장미희9, 장미희10번}과 S2={장미희4, 장미희6번}는 내용이 모두 동일한 문서들이다. 기사의 출처를 보면 동아일보, 한국일보, 세계일보, 연합세계는 모두 연합뉴스의 기사를 그대로 기사로 작성한 것을 알 수 있고, JOINS와 중앙일보는 동일한 회사지만 회사 이름을 한글과 영

표 3 구글 뉴스에서 '장미희' 키워드를 통해 찾은 기사들의 출처

문서 이름	신문사
장미희1	한국경제
장미희2	조선일보
장미희3	동아일보
장미희5	매일경제
장미희7	조인스연예
장미희8	한국일보
장미희9	세계일보
장미희10	연합세계

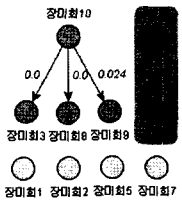

그림 4 구글 뉴스에서 '장미희'키워드를 통해 찾은 기사들의 표절 유사도 그래프

어로 바꿔가며 기사를 재전송하여 구글 필터링을 피한 것을 알 수 있다.

만약 구글 뉴스 검색 시스템에 DEVAC 시스템을 사용하게 된다면 '장미희'키워드에 대한 결과 검색은 '장미희1', '장미희2', '장미희4', '장미희5', '장미희7', '장미희10' 이렇게 6개 문서를 보여주게 될 것이다. 포털 사이트가 DEVAC 시스템을 이용한다면 동일문서를 재전송하는 언론사들의 과도한 검색작업을 방지할 수 있고, 검색결과의 신뢰성을 높일 수 있을 것이다.

5.2 리포트 표절 검사

일반적으로 한 학급의 학생들이 제출한 n개의 과제를 한 명의 교사가 검사하며, 교사는 과제들 사이의 표절을 탐색하기 위해 $O(n^2)$만큼의 비교 과정을 거쳐야 한다. 혼자 힘으로 모든 과제에 대해 이와 같은 검사를 하는 것은 거의 불가능하다. 표 4는 실제 한 학급의 독후감 과제 55개를 모아 표절 탐색을 수행한 후, 유사도 값이 큰 두 과제에 대해 표절 영역을 나타낸 것이다. 두 표절 영역을 비교하여 살펴보면, 중간 중간에 새로운 단어가 삽입되거나 삭제된 경우가 있으며, 어절의 순서를 바꾼 경우도 찾을 수 있다. 하지만 두 문장을 읽어보면 매우 유사하며 의미도 동일하다. DEVAC 시스템은 표절 행위자가 삽입, 삭제, 문장의 도치등과 같이 여러 가지 방법을 사용하여 교묘하게 표절을 행하여도 표 4와 같이 정확하게 찾아낸다.

표 5는 여러 종류의 실험 데이타를 이용하여 시스템의 탐색 시간을 측정한 결과이다. 한 학급의 과제들을 탐색하는 데에는 750초가량 걸리며 소설 11권 사이의 표절을 탐색하는 데에는 128초정도 걸렸다.

표 4 두 독후감을 표절 탐색하여 찾아낸 표절 영역

문서1	문서2
[illegible]	[illegible]
[illegible]	[illegible]
[illegible]	[illegible]
[illegible]	[illegible]
[illegible]한다	[illegible]했다 그래서
[illegible]사는	[illegible]은 그의
[illegible]인	[illegible] 중 가장 뛰어난 제자인
[illegible]	[illegible]
[illegible]러 보낸다 [illegible]	[illegible]게 하였다 [illegible]
[illegible]도 [illegible]	[illegible]이 [illegible]
보내 모처럼의 [illegible]	그의 [illegible]
[illegible]한다	[illegible]하기 위해

표 5 다양한 데이타에 따른 탐색 시간 측정

	문서 수	평균크기 (어절)	비교 횟수	시간 (sec)
뉴스	12	154	132	1
독후감	55	922	2,970	750
소설	11	73,582	110	128
자기소개서	11,029	118	121,627,812	16,276

6. 결 론

본 논문은 한글문서의 표절탐색에 적합한 표절탐색 시스템인 DEVAC 시스템을 제안하였다. DEVAC 시스템은 'k-mer' 방식과 '지역정렬' 방법을 기반으로 하며 대용량의 문서에서도 빠르고 정확하게 표절 탐색을 수행한다.

우리는 한글 말뭉치로 다양한 실험을 하여 한글 문서의 특성을 찾았다. 표절이 없는 일반적인 문서에서 나타나는 유사도를 이용하여 확률 모델을 만들었으며 이 모델을 이용하여 객관적인 표절 판정 방법을 제시하였다. 또한 시스템의 specificity와 sensitivity를 측정하여 최적 파라미터를 구하였다. 구글 뉴스 필터링과 리포트 표절 검사 실험을 통해 DEVAC 시스템의 다양한 이용 가능성을 보였으며 시스템의 성능을 증명하였다. 현재 DEVAC 시스템은 웹 서비스 중에 있으며, 사이트(http://devac.cs.pusan.ac.kr:8080/)에서 회원 가입을 통해 사용할 수 있다.

참 고 문 헌

[1] Turnitin. http://www.turnitin.com/

[2] Donaldson, J. L., Lancaster, A., and Sposato, P. H. A plagiarism detection system. In Proceedings of the Twelfth SIGCSE Technical Symposium on Computer Science Education. 21-25, 1981.

[3] CloneChecker: A Software Plagiarism Detector. http://ropas.snu.ac.kr/n/clonechecker/

[4] Geoff Whale. Plague: Plagiarism detection using program structure. Department of Computer Science, University of New South Wales, May 1988.

[5] David Gitchell and Nicholas Tran. Sim: a utility for detecting similarity in computer programs. In SIGCSE '99: The proceedings of the thirtieth SIGCSE technical symposium on Computer science education, 266-270, 1999.

[6] Wise. YAP3: Improved detection of similarities in computer program and other texts. SIGCSEB: SIGCSE Bulletin, 28, 1996.

[7] Ryu Chang-Keon, Kim Hyong-Jun, Park Soo-Hyun, and Cho Hwan-Gue. DEVAC(Document EVolution Analyzing Center). http://devac.cs.pusan.ac.kr:8080/

[8] Schleimer, S., Wilkerson, D. S., and Aiken, A. Winnowing: local algorithms for document fingerprinting. In Proceedings of the 2003 ACM SIGMOD international Conference on Management of Data. 76-85. June 09-12, 2003.

[9] Cameron, M., Williams, H. E., and Cannane, A. Improved Gapped Alignment in BLAST. IEEE/ACM Trans. Comput. Biol. Bioinformatics 1, 3, 116-129. Jul. 2004.

[10] Leslie, C. and Kuang, R. Fast String Kernels using Inexact Matching for Protein Sequences. J. Mach. Learn. Res. 5, 1435-1455. Dec. 2004.

[11] 21세기 세종계획. http://www.sejong.or.kr/