문장-질의 유사성을 이용한 웹 정보 검색의 성능 향상

(Performance Improvement of Web Information Retrieval Using Sentence-Query Similarity)

박 의 규 † 나 동 열 †† 장 명 길 †††
(Eui-Kyu Park) (Dong-Yul Ra) (Myung-Gil Jang)

요 약 인터넷의 발전으로 웹 상에 수많은 문서 및 정보가 존재하는 상황에서 사용자가 원하는 정보를 담은 웹 문서를 검색하여 주는 웹 정보 검색 기술은 매우 중요하게 되었다. 본 논문에서는 웹 정보 검색 시스템의 성능 향상에 효과적인 몇 가지 주요한 기술을 제안하였다. 기존 시스템들은 주로 문서와 질의의 유사도를 계산하여 이를 주요 정보로 이용하였다. 그러나 본 논문에서는 여기에서 한 걸음 더 나아가 문서 안의 각 문장들이 질의와 얼마나 유사한가를 계산하여 이를 이용하는 기법을 제안하였다. 이러한 문장-질의 유사도를 성숙된 자연어 처리 기술 없이 근사적으로 계산하는 방법을 소개하였다. 그리고 이 계산 작업은 문서 수의 증가에 선형적인 계산량의 증가를 가져 옴을 보임으로써 실용적인 대용량 시스템에서도 사용할 수 있음을 보였다. 그 다음으로 제안된 주요한 기술은 출력 문서의 순위화에 계층적인 개념을 도입하는 것이다. 이 기법을 사용함으로써 상당한 성능 향상을 이룰 수 있음을 보였다. 그 외에도 웹 문서의 특징인 하이퍼 링크 정보와 타이틀 정보를 이용하여 어느 정도의 성능 개선을 가져올 수 있음을 보였다. 이러한 기술들의 타당성을 입증하기 위해 대용량 웹 정보검색 시스템을 개발하고 실험하였다.

키워드 : 웹, 정보검색, 문장-질의 유사도, 계층화, 하이퍼 링크, 앵커텍스트

Abstract Prosperity of Internet led to the web containing huge number of documents. Thus increasing importance is given to the web information retrieval technology that can provide users with documents that contain the right information they want. This paper proposes several techniques that are effective for the improvement of web information retrieval. Similarity between a document and the query is a major source of information exploited by conventional systems. However, we suggest a technique to make use of similarity between a sentence and the query. We introduce a technique to compute the approximate score of the sentence-query similarity even without a mature technology of natural language processing. It was shown that the amount of computation for this task is linear to the number of documents in the total collection, which implies that practical systems can make use of this technique. The next important technique proposed in this paper is to use stratification of documents in re-ranking the documents to output. It was shown that it can lead to significant improvement in performance. We furthermore showed that using hyper links, anchor texts, and titles can result in enhancement of performance. To justify the proposed techniques we developed a large scale web information retrieval system and used it for experiments.

Key words : web, information retrieval, sentence-query similarity, stratification, hyper link, anchor text

† 비 회 원 : 연세대학교 전산학과
ekpark63@dragon.yonsei.ac.kr
†† 종신회원 : 연세대학교 전산학과 교수
dyra@dragon.yonsei.ac.kr
††† 비 회 원 : 한국전자통신연구원 지식마이닝연구팀
mgjang@etri.re.kr
논문접수 : 2003년 8월 20일
심사완료 : 2005년 4월 7일

1. 서 론

웹 문서 검색은 현대 사회에서 매우 중요한 기술이다. 현재 인터넷 웹 상에는 수많은 정보가 존재하고 있다. 정보의 종류 및 문서의 수도 방대하다. 그러나 이러한 많은 정보의 바다가 있음에도 불구하고 사용자가 원하는 정보를 손쉽게 찾지 못한다면 그 많은 정보도 별 소용이 없게 된다. 이런 점에서 웹 문서 검색 기술은 매우

중요하게 되었고 앞으로 웹의 정보의 양이 점점 증가할 것으로 예상되는 미래에서는 더욱 그러하다.

정보검색과 관련되는 기술의 발전을 위해서 미국 NIST(National Institute of Standards and Technology)의 주관으로 TREC 학술대회가 매년 열리고 있다[1]. 여기에는 여러 기술 분야별로 전문화된 트랙(track)을 두어 기술 발전을 유도하고 있다. 웹 검색과 관련된 기술은 웹 트랙에서 담당하고 있다. 그런데 TREC 에서는 웹 문서의 검색 문제와 관련하여 원래는 전통적인 정보검색 문제와 같은 관점을 가진 애드혹(ad-hoc) 문제를 다루었다. 그러나 웹 정보 검색이 점차 보편화되고 광범위한 사람들이 이용하여 그 중요성이 커지면서 여러 가지 다른 검색 방식들이 나타나게 되었다. 먼저 홈페이지 검색 작업(home page finding task)이 등장하게 되었는데[2], 이것은 질의가 원하는 정보를 담고 있는 문서를 가진 웹 사이트(site)의 시작(entry) 페이지를 질의에 대한 답으로 내주는 것이다. 이를 이 사이트의 홈페이지라 부른다. 이 문제와 관련된 연구로는 [3]이 있다.

그러나 TREC에서는 최근에는 홈페이지 검색 작업의 진보된 형태인 토픽증류 작업(topic distillation task)과 지정페이지 검색 작업(named page finding task)에 관심을 보이고 있다[4]. 토픽증류 작업은 과거의 홈페이지 검색 문제의 약간 진보된 형태로서 질의가 요구하는 정보를 포함한 웹 페이지로 도달할 수 있는 길목이 되는 웹 페이지 즉 키 리소스(key resource) 페이지를 찾아주는 문제이다. 반면 지정페이지 검색이란 질의가 원하는 정보를 포함하고 있는 웹 페이지나 문서를 찾아 주는 것이다. 본 논문에서는 지정페이지 검색 작업을 목표로 하는 시스템에 대한 것이다. 우리가 이 문제를 먼저 다루는 이유는 이것이 보통 사용자가 가장 많이 사용하는 검색 모드라 생각되기 때문이다.

정보검색 시스템의 실험을 위해서는 테스트컬렉션(test collection)이 있어야 한다. 이것은 웹 문서 집단, 질의 집합 및 각 질의에 대한 정답으로 이루어져 있다. 테스트컬렉션이 필요한 이유는 시스템의 성능을 객관적으로 평가할 수 있기 때문이다. 이를 이용하여 다른 시스템과의 객관적인 성능 비교가 가능하다. 정보검색 기술의 발전을 위해서는 대용량 테스트컬렉션이 있어야 하며 이의 준비에는 방대한 노력과 비용이 필요하므로 이의 준비는 쉽지 않다. 미국 NIST에서는 정보검색 기술의 발전을 위해서 대용량 테스트컬렉션을 제공하며 매년 TREC 학술대회를 개최하여 연구 집단들이 개발한 시스템의 성능 비교 및 정보교환이 가능하도록 하고 있다. 이와 유사한 것으로 일본에서도 NTCIR[5] 이라는 학술 대회를 매년 개최하고 있으며, 동양어권 정보검색 평가대회로 확대하고 있다.

본 논문에서 소개하는 기술은 TREC의 웹 트랙에 참가하기 위해 개발된 시스템의 경험을 바탕으로 하고 있다. 이 시스템은 NIST에서 만든 TREC의 웹트랙(web track)을 위한 테스트 컬렉션을 이용하였다. 먼저 문서 집단은 10GB 이상에 달하는 대량의 웹 문서 집단으로서 NIST의 의뢰로 호주의 CSIRO에서 준비한 문서 집단이다. 이의 구축에 대한 자세한 점은 [6]에서 알아 볼 수 있다. 질의는 NIST에서 준비한 것으로 총 150 개의 질의를 가지고 있다. 각 질의에 대한 답은 주어진 문서 집단에 존재하는 문서에 한한 것으로서 NIST에서 준비한다. 수백만 문서 집단에서 정답을 찾는다는 것은 매우 어려운 작업이다. 이를 위해서 NIST는 pooling이라는 기법을 사용하고 있다[7].

본 논문에서는 우리가 개발한 시스템이 이용하는 기술을 소개하고 자 한다. 본 시스템이 가진 가장 큰 특징의 하나로는 우선 문장-질의 유사성이라는 새로운 개념을 도입하여 이용함으로써 많은 성능 향상을 가져온 점이다. 그리고 웹 문서의 특징을 이용하기 위하여 웹 문서에 포함된 하이퍼 링크 정보를 적절히 이용하였다. 시스템 성능의 향상을 위해서 그 밖에도 제목 정보의 이용, 문장-질의 공통색인어 수에 의한 검색결과의 계층화 등 다양한 기법을 이용하고 있다. 본 시스템은 당장의 상용화를 목표로 개발된 시스템은 아니지만 대량의 문서 집단에 대한 색인 검색을 수행하여야 하므로 효율적인 색인 구조 및 검색 속도의 향상을 위해 고려한 점들이 있으며 이에 대하여도 언급하고자 한다.

본 논문의 구성은 다음과 같다. 2장에서 본 연구에서 개발된 웹 검색 시스템의 바탕을 이루는 기본 모델에 대하여 기술한다. 그리고 이 시스템이 이용하는 가장 특징적인 기법인 문장-질의 유사도의 이용에 관한 내용이 3장에 주어지며, 웹 문서의 특징인 하이퍼 링크가 제공하는 정보의 이용 기법이 4장에서 설명된다. 5장은 기타 몇 가지의 성능 향상 기법이 소개된다. 6장에서 실험 및 그 결과를 보여 주며, 7장에서 결론을 기술한다.

2. 기본 시스템의 구성

2.1 기본 모델

우리 시스템은 기본적으로 벡터공간(vector space) 모델을 기반으로 하고 있다[8,9]. 본 논문에서 사용하는 각종 기호를 정리하면 다음 표 1과 같다.

이 모델에서 문서는 t 차원의 벡터로 표현된다. 즉 문서 d_j는

$$d_j = (w_{1,j}, w_{2,j}, w_{3,j}, \ldots, w_{t,j}) \quad (1)$$

여기서 i 번째 원소 $w_{i,j}$는 색인어 k_i에 대응되는 값으로서 문서 d_j에 대하여 이 색인어가 가지는 가중치로 해석할 수 있다.

표 1 여러 기호 및 설명

t : 전체 색인어 수	df_i : 색인어 k_i가 나타난 문서의 수
N : 전체 문서수	$tf_{i,j}$: 색인어 k_i의 d_j 에서의 빈도수
Ω : 전체 문서 집합	$tf_{i,q}$: 색인어 k_i의 질의 q 에서의 빈도수
dj : Ω안에서 j 번째 문서	$f_{i,j}$: 색인어 k_i 의 문서 d_j 에서의 정규화된 빈도수
ki : 전체색인어 집합에서 i 번째 색인어	L : 앵커 텍스트

$$f_{i,j}=\frac{tf_{i,j}}{tf_{\max,j}} \text{ where } tf_{\max,j} = MAX_i\, tf_{i,j}$$

$$w_{i,j} = f_{i,j}\times \log\frac{N}{df_i}$$

질의 q에 대한 표현은

$$q = (q_1, q_2, \ldots, q_t) \quad (2)$$

로서, 색인어 k_i에 대응되는 원소 q_i는

$$q_i = (0.5+\frac{0.5tf_{i,q}}{tf_{\max,q}})\times \log\frac{N}{df_i}$$

벡터공간 모델에 기반한 검색 기법으로 우리는 식 (3)에서와 같은 코사인(cosine) 계수값을 문서와 질의 사이의 유사도 sim_0로 이용한다.

$$sim_0(d_j, q)=\frac{d\cdot q}{|d|\times|q|} \quad (3)$$

위 식에서 분자는 두 벡터의 내적(inner product)를 나타내며, 분모에서 ||는 벡터의 길이를 나타낸다.

검색의 기준으로 사용되는 값을 보통 검색기준값(RSV; retrieval status value)이라 부르며 기본 모델만을 이용하는 시스템에서는 식 (3)의 sim_0만을 이용하게 된다. 즉,

$$RSV(d, q) = sim_0(d, q) \quad (4)$$

검색을 위해서 시스템은 질의 q가 주어지면 모든 문서 d에 대하여 $RSV(d,q)$를 계산한다.[1] 검색결과는 RSV가 0보다 큰 모든 문서의 순위화된 집합(ranked list)이다. RSV가 큰 문서일수록 앞 순위가 되며 순위(rank) 값은 작게 된다.

2.2 색인구조

색인작업이란 문서에서 나타나는 색인어에 관한 정보를 색인 저장구조에 저장하는 것이다. 본 시스템에서는 명사, 동사, 형용사의 품사를 갖는 모든 단어를 색인어로 한다. 단, 단어의 원형(stem)을 구하여 이를 색인어로 이용한다.

색인 저장구조는 역화일(invert file)을 사용한다. 즉 임의의 색인어에 대하여 이 색인어가 출현한 문서들을 추출할 수 있도록 저장하는 것이다. 본 시스템에서는 속도 향상을 위하여 B-tree로 역화일을 구현한다. 그림 1에서 각 색인어에 대하여 B-tree에 색인어노드 하나씩이 존재한다. 색인어노드에는 이 색인어가 출현한 문서의 수인 df(document frequencey)가 저장된다.

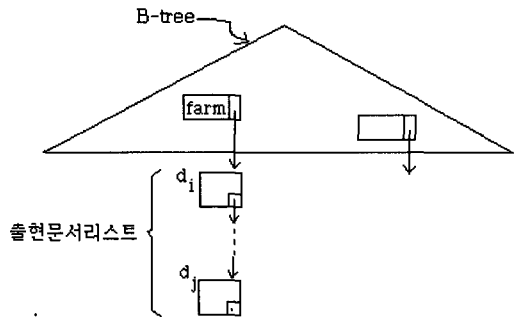

그림 1 색인 저장 구조.

색인어 노드는 이 색인어가 출현한 문서를 저장하는 노드들을 가진 출현문서리스트를 가리킨다. 출현 문서리스트의 각 문서노드에는 문서 번호 뿐만 아니라 해당 색인어가 해당 문서에 나타난 색인어출현횟수 tf(term frequency)를 저장한다.

각 색인어에 대한 출현문서리스트가 연결리스트(linked list) 형태로 저장되어 있는 경우 처리 속도가 크게 저하된다. 그 이유는 문서 노드 하나를 액세스 할 때마다 디스크 탐색이 일어나는데 문서의 수가 수천, 수만 이상이 되는 경우가 많으므로 많은 시간이 소요된다. 이를 해소하기 위해서 그림 2에 나타난 것처럼 색인어에 대한 출현 문서 리스트(document occurrence list; DOL)를 배열형태로 저장하는 포스팅 화일을 이용한다. 이 경우 한번의 디스크 읽기에 의하여 해당 색인어에 대한 출현문서리스트 안의 모든 문서노드를 한번에 가져올 수 있어 시간절약 효과가 크다. 그러나 이러한 그림 2의 포스팅 파일을 포함하는 색인구조는 먼저 그림 1과 같은 색인구조를 모두 만든 후에 그 결과를 통하여 생성하여야 하므로 색인 단계에서의 시간은 증가된다.

대량의 문서집단에 대하여 색인 작업을 하다 보면 어느 시점에서부터 갑자기 색인속도가 급격히 떨어진다. 여기서의 색인 속도란 한 문서에 출현한 한 색인어에 대하여 이에 대한 문서노드의 생성 및 저장을 위한 시간을 말한다. 본 시스템 개발의 경우 B-tree가 100만개의 색인어를 저장할 쯤부터 이러한 현상이 발생하기 시작하였다. 이 뒤에는 처리 속도가 급격히 감소함이 관찰되었다. 이 문제에 대한 대처 방안으로 우리는 전체 색

1) 실제로는 유사도가 0보다 큰 문서만을 고려하는 기법을 사용하여 속도를 높인다.

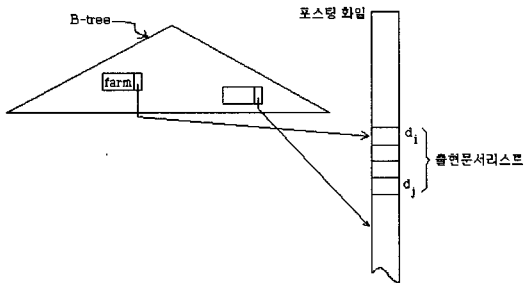

그림 2 속도 향상을 위한 색인 구조.

인 구조를 여러 개의 B-tree에 나누어 저장하는 기법을 사용한다. 현재로서는 알파벳을 기준으로 특정 구간에 속하는 단어들은 그 구간에 대한 B-tree에 데이타를 저장하는 것을 말한다.

3. 문장-질의 유사도의 이용

위에서 설명한 검색 모델 즉 전통적인 벡터공간 모델은 문서와 질의 사이의 유사도를 계산하는 데 있어서 질의와 문서가 얼마나 색인어를 공통으로 가지고 있는가를 이용하는 것이다. 이것은 문서와 질의 사이의 유사성을 이용하는 것이다.

그러나 우리는 질의와 문장 사이의 유사도 정보도 중요하다고 생각 한다. 이 문제를 표 2의 예를 통하여 생각해 보자.

표 2 문장-질의 유사도를 위한 예

d_i	시카고의 자연사 박물관은 매우 유명하다. 가족과 함께 여행을 온 필라델피아에서 온 학생이 그 크기에 놀랐다. 여기에는 .....
d_j	존은 펜실베니아 대학 캠퍼스를 둘러 본 후 필라델피아에 있는 박물관을 방문하였다. 거기에서 미국 문화를 알려 주는 많은 물건을 볼 수 있었는데 특히 .....
q	필라델피아 박물관은?

위 두 문서를 포함한 문서집단에 대하여 주어진 질의를 수행한다고 하자. 이 질의에 대하여 기존의 방법에 의하면 q와 d_i 사이의 유사도는 q와 d_j와의 유사도와 비슷할 것이다. 왜냐하면 두 문서 모두 질의가 가진 색인어 "필라델피아"와 "박물관"을 포함하고 있기 때문이다.[2] 그러나 사람이 판단한다면 분명히 d_j가 더 관련성이 높은 것으로 보인다. 그 이유는 d_j의 첫 번째 문장 때문이다. 이 문장의 의미에 질의가 요구하는 정보가 있음을 알 수 있다. 물론 이것은 문장의 의미를 이해할 수 있는 경우에 가능하다.

이상적인 정보검색이란 문장의 의미를 이해하여 문서 안에 질의가 원하는 정보가 있는 지 판단하여 결정하는 방식일 것이다. 그러나 현재의 자연어 처리 기술은 이러한 이상적인 정보검색을 가능하게 할 정도로 발전하지 못하였다[10]. 구 단위 색인이나 의미단위 색인을 추구하는 연구가 있으나 현재로는 실용화 단계까지 이르지는 못하고 있다.

지정페이지검색에서는 하나의 질의가 긴 자연어 문장들로 구성되지 않고, 단지 몇 개의 단어 열로 구성된다. 그리고 현재의 웹 검색에서 대부분의 질의 입력은 이러한 모습을 띄고 있다. 여기서 우리는 다음과 같은 가정을 하고자 한다: "질의가 나타내는 의미는 거의 하나의 문장 안에 존재한다." 다시말하면 질의의 의미가 문서 안에서 여러 문장으로 분산되어 나타나는 것이 아니라 대부분 한 문장 안에서 표현된다. 따라서 질의의 의미와 매칭되는 문장을 찾을 수 있다면 이 문장을 포함한 문서는 질의와의 관련성이 높다고 말할 수 있다.

이 아이디어를 적용하기 위해 우리는 문서 내에 질의가 나타내는 의미를 가진 문장이 있으면 그 문서의 질의에 대한 관련도를 증가시키는 방법을 취하고자 한다. 이것은 결국 문장과 질의와의 의미의 유사성을 측정하여 이를 검색 점수에 반영하는 방식으로 구현될 수 있다. 현재로서의 문제점은 문장의 구문분석, 의미해석을 통한 유사성 측정은 어렵다는 것이다. 이 문제에 대한 우회적이고 단기적인 대책으로 문장과 질의의 색인어 공유 정도를 이용하고자 한다. 문장 s와 질의 q의 유사도를 $C(s,q)$라 하면, 이를 s와 q의 색인어의 공유정도라는 근사치로 구한다. 앞으로 자연어 처리 기술이 발달하면 더 정교한 기법에 의한 계산을 이용할 수도 있다.

$$C(s,q)=\begin{cases}\left(\dfrac{|s\cap q|}{|q|}\right)^{k} & \text{if } |s\cap q|\geq\tau(|q|)\\ 0 & otherwise\end{cases} \quad (5)$$

|s ∩ q|란 문장 s와 질의 q에 같이 나타나는 색인어 집합의 원소수를 나타낸다. |q|는 질의가 가진 색인어의 수를 나타낸다. 지수 k는 문장 내에 나타나는 질의 색인어 수의 중요성 정도를 조절하기 위한 파라메타이다. k가 증가할수록 공유 색인어의 수가 많은 경우가 적은 경우보다 더 중요하게 작용하게 된다. $\tau(|q|)$는 질의가 가진 색인어 수에 따라 문장이 공유하여야 하는 색인어 수가 특정값 이하이면 문장 유사도의 기여를 없애기 위한 것이다. 현재는 τ의 값이 다음처럼 정해져 있다:

$\tau(1)=2$, $\tau(2)=1$, $\tau(3)=2$, $\tau(4)=2$, $\tau(5)=2$, and $\tau(i)=3$ for $i\geq 6$.

문서 안의 모든 문장에 대하여 문장-질의 유사도를 합하면 문서와 질의의 유사도를 얻는다. 즉,

$$sim_1(d,q)=\sum_{i=1}^{n} C(s_i,q) \text{ where } si\text{'}s \text{ are sentences in } d. \quad (6)$$

2) 문서 벡터의 길이가 약간의 영향을 줄 수 있는 정도이다.

우리는 이 유사도를 문서의 관련도 *RSV* 계산의 한 부분으로 이용하고자 한다. 다시 위 표 [2]의 예를 보면 문서 d_j의 두 번째 문장 s'은 질의 안의 색인어를 모두 포함하므로 $sim_1(s', q)$는 다른 문장들에 비해서 높은 값이 될 것이며 이에 의하여 문서 d_j가 d_i보다 검색점수가 높아지게 되는 것이다. 문서 d_i에서는 질의 색인어가 다른 문장에 흩어져 나타나나 d_j에서는 같은 문장에 나타났다. 여기에서 다음과 같은 사실을 관찰할 수 있다: 질의가 가진 색인어들이 문서 안에서 여러 문장에 흩어져 나타나는 것보다는 같은 문장 안에 모여서 나타나는 것이 좋다.

기본 모델에 문장-질의 유사도까지 이용한 시스템의 검색기준값은 다음과 같이 계산된다. 여기서 $sim_0(d, q)$는 기본적인 검색 모델에 의한 문서-질의 관련도를 나타내며 앞 장에서 구하였다.[3)]

$$RSV(d,q) = sim_0(d,q) + \alpha \cdot sim_1(d,q) \qquad (7)$$

위 식에서 계수 α는 문서 검색기준값에서 문장-질의 유사성이 기여하는 정도를 조절하기 위한 가중치이다.

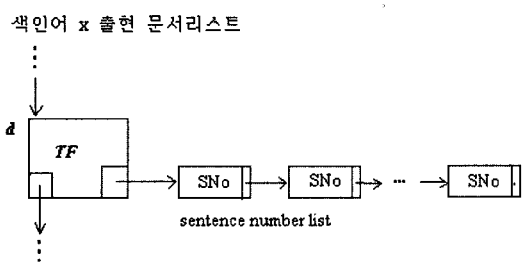

그림 3 문장 번호 저장을 위한 색인 구조.

기존의 정보검색 시스템이 이용하는 색인구조에서는 색인어(x라 하자)마다 x가 출현한 문서들이 무엇들이며 이들 각 문서마다 x가 몇 번 나왔는지의 횟수 즉 색인어빈도 (tf)를 저장하고 있다(그림 3 참조). 문장-질의 유사도를 계산하기 위해서는 이러한 정보 이외에 문서마다 그 문서 안의 x가 나타난 문장마다 문장 번호를 저장하여야 한다. 예를 들어 x가 문서 d에 출현하였다 하면 그림 3과 같이 문서 d에 대한 문서노드의 문장 번호 리스트(sentence number list; SNL)에 이러한 문장 번호들이 저장된다.

본 절에서 설명한 문장 질의 유사도 기법을 이용하기 위해서는 기존의 시스템에 비하여 여러 가지 오버헤드를 수반하게 된다. 먼저 색인과정에서의 오버헤드에 대하여 살펴 보자. 색인 과정에서 각 색인어마다 이것이 출현한 적이 있는 문서에 대하여 이를 해당 문서출현리스트(document occurrence list; DOL)[4)]의 한 노드로 매달게 된다. 이를 문서노드라 한다. 주어진 색인어가 출현한 문서 안의 문장 번호를 저장하는 작업이 추가로 소요되는 오버헤드인데 이것은 바로 문장번호리스트 즉 SNL을 문서 노드에 준비하는 작업이다. 이 작업은 시간적으로는 단지 문장 번호들을 SNL에 넣는 것으로서 색인어의 문서 내의 빈도수 (tf)에 비례하는 시간으로서 다른 작업에 비하여 작업량이 크지 않다. 시스템이 다루는 문서의 수가 증가할 때 tf는 이와 관련한 증가는 없으나 DOL의 길이가 증가하게 되고 이에 비례하여 SNL의 준비 시간도 증가한다. 즉 시간적인 오버헤드는 전체 문서 수를 n이라 하면 $O(c_1 n)$으로서, 이는 기존 시스템이 수행하는 다른 작업의 최적인 상황에서의 시간 복잡도인[5)] $O(c_2 n)$과 같다(c_1, c_2은 작업과 관련된 상수). 결국 문서수가 증가하여도 전체 처리 시간에 대한 문장-질의 유사도 처리에 필요한 계산량의 비중은 $c_1/(c_1+c_2)$으로서 큰 변화가 없다는 것을 의미하며, 문서 수의 증가에 대한 계산 능력(computing power)의 대처에서 기존 시스템의 경우와 같이 하면 된다는 것을 의미한다.

색인에 소요되는 공간적인 오버헤드를 보면 이 역시 출현문서리스트(DOL)의 문서노드마다 tf에 비례하는 만큼의 노드가 SNL에 저장된다. 보통 tf는 작은 수로서 총 문서 수 n과는 관련이 없다. 결국 시간적인 복잡도와 마찬가지로 오버헤드의 공간적인 복잡도도 O(n)이다. 따라서 총 문서 수 n에 대한 선형적인 관계로서 큰 문제가 되지 않는다. 예를 들면 우리가 개발한 시스템의 경우 색인 저장 구조[6)]의 10% 정도를 SNL의 저장에 사용하고 있다. 전체 문서 수가 1백 20만 개인 본 연구에서 개발한 시스템의 경우 4GB의 색인 저장 구조 중 350MB를 문장번호의 저장에 사용하고 있다. 여기서 중요한 점은 시스템의 문서 수가 증가하여도 이 비중은 변하지 않는다는 점이다. 만약 1억 2000만 개의 문서를 다루는 시스템이 된다면 전체 색인저장구조 400GB 중 역시 약 10%인 35GB를 SNL의 저장에 사용하게 된다. 따라서 색인과 관련하여서는 문서 수의 증가에 대하여 기하급수적이거나 높은 차원의 다항식에 따른 증가가 아니므로 색인 시간 오버헤드의 경우처럼 큰 문제가 되지 않는다.

다음으로 검색 시간의 오버헤드를 살펴 보자. 문장-질

3) 주: sim0(d,q) 는 벡터공간 모델이외의 다른 검색모델에 의해서 계산된 값을 이용해도 된다.

4) 그림 1 참조.

5) 시스템의 최적의 시간 복잡도는 최소한 O(n) 으로 볼 수 있다. 만약 시작 복잡도가 O(c) 즉 상수인 경우는 문서 수가 증가해도 같이 시간이 필요하다는 것을 의미하는데 이는 현실적으로 불가능하다.

6) 여기서의 색인저장구조는 문서를 저장하는 공간은 제외한 색인구조만을 말한다.

의 유사도와 관련하여 검색 시에 수행하는 주요 추가 작업은 SNL에 저장된 문장번호를 처리하는 작업이다. 문서 d와 질의 q의 유사도를 계산하는 경우를 보자. 질의 q가 가진 색인어들의 DOL에서 문서 d를 찾아야 하는데 이는 기존의 시스템도 마찬가지로 수행하는 작업이다. 찾아진 문서 노드들의 SNL들의 문장 번호의 중복 카운트를 문장 번호마다 계산하는 것이 수행할 주요 작업이다. 그러나 이 작업은 매우 간단히 수행될 수 있다. 가장 긴 문서의 문장 수만큼의 원소 수를 가진 배열(array)을 준비한다. SNL에 저장된 각 문장 번호 마다 이 번호에 해당하는 배열의 원소의 값을 1씩 증가한다. 이 작업을 모든 SNL에 대하여 수행한다. 결국 SNL의 길이는 tf에 비례하므로 질의문의 길이가 |q|라면 소요시간은 O(|q|·tf)와 같다. 이 작업은 DOL의 문서 수만큼 수행하여야 하는데 이 길이는 총 문서수 n에 비례한다. 결국 추가로 소요되는 검색 작업 시간의 오버헤드는 O(|q|·tf·n)가 되며 |q|, tf는 n의 증가와 관련이 없으므로 O(n)이 된다. 결국 색인 시간의 오버헤드에서 설명하였듯이 문서량이 증가하더라도 큰 문제가 되지 않는다.

지금까지 설명한대로 문장-질의 유사도 기법을 사용하는데 있어서 시간적 공간적 오버헤드는 총 문서 수의 증가가 있더라도 선형적인 증가로서 큰 문제가 되지 않는 것으로 분석된다. 실제 상용 웹 서치 엔진에서 가장 문제가 되는 것은 방대한 문서량의 증가이다. 본 기법은 이런 상황에도 계산 능력의 적절한 증가를 유지한다면 적용될 수 있는 실용적인 기술이라 생각된다.

4. 링크 정보 이용 기법

4.1 접근 방법

웹 문서 검색에 대한 연구의 초기에서부터 가장 중요하게 간주된 것이 웹문서가 포함하는 하이퍼링크 정보를 이용하고자 하는 것이었다[11]. 대부분의 연구에서 링크 정보의 이용은 링크의 연결관계 및 연결 관계를 맺는 문서가 가진 점수의 이용이었다. 그러나 초기의 많은 연구에서 밝혀진 바는 링크 정보의 이용이 검색 성능의 향상에 큰 기여를 하지 못한다는 것이었다[12].

그러나 본 논문에서는 지금까지 이용된 여러 복잡한 링크 이용 방식과 달리 링크 정보의 이용을 다음과 같이 단순화하였다. 즉 링크의 앵커텍스트(anchor text)를 이용하는 것이다. 예를 들어 설명하자. 그림 4에서 링크 l은 문서 d_i에서 출발하며 d_j를 가리킨다. 앵커로 이용된 텍스트는 "의료 보험" 이다.[7] 그렇다면 이 링크의 의미는 의료보험에 대하여 알고 싶으면 이 링크가 가리키는 문서(여기서는 d_j)를 방문해 보라는 의미이다. 즉 문서 d_j는 의료 보험에 대한 주제를 다루고 있다는 의미이다.

따라서 문서 d_j의 검색기준값 계산에 이 문서를 가리키는 링크들의 앵커텍스트를 이용하고자 하는 것이 우리의 접근 방법이다. 우리는 이를 다음 두가지 방식으로 구현하고자 한다.

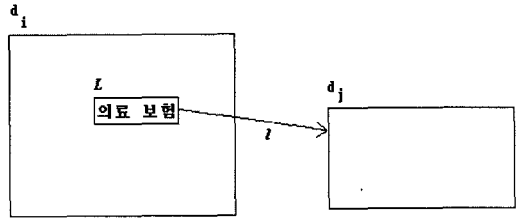

그림 4 링크와 앵커텍스트

• **방법 1** : 이 방법은 앵커텍스트와 질의와의 관련도를 코사인(cosine) 계수를 이용하여 구하는 것이다. 문서 d를 가리키는 링크들의 앵커텍스트들을 L_1, ..., L_m이라 하자. 여기서 각 L_i를 벡터표현으로 만들어 질의와의 코사인 계수 값을 이용하여 유사도를 구한다. 즉,

$$sim_{2a}(d, q) = \sum_{i=1}^{m} \frac{L_i \cdot q}{|L_i| \times |q|} \quad (8)$$

• **방법 2** : 이 방법은 앵커텍스트와 질의와의 관련도를 색인어 공유정도를 이용하여 구하는 것이다. 즉 앞에서 소개한 문장-질의 유사도와 관련하여 소개된 C 함수를 이용한다.

$$sim_{2b}(d, q) = \sum_{i=1}^{m} C(L_i, q) \quad (9)$$

링크 정보를 이용함으로써 기여되는 점수는 다음과 같다.

$$sim_2(d, q) = sim_{2a}(d, q) + sim_{2b}(d, q) \quad (10)$$

링크에 의한 정보를 모두 이용한 문서의 검색점수는 다음과 같다.

$$RSV(d, q) = sim_0(d, q) + \alpha \cdot sim_1(d, q) + \beta \cdot sim_2(d, q) \quad (11)$$

상수 β는 링크정보에 의하여 제공되는 유사도의 가중치를 조절하기 위한 계수이다.

4.2 링크정보 이용의 효율적인 구현

위 절에서 소개한 링크 정보 이용 방법은 매우 효율적으로 구현하지 않으면 시스템의 속도를 매우 저하시킬 가능성이 있다[13]. 기본모델을 설명하는 장에서도 언급하였듯이 sim_0 값의 계산 시에 이 값을 실제로 가지는 문서들만을 처리함으로써 속도의 향상을 꾀하였다. 마찬가지로 속도 문제 때문에 모든 문서에 대하여 sim_2 값을 계산하여 볼 수는 없다. sim_2 값이 0인 문서는 가능하면 고려 대상에서 제외하여야 한다.

7) 앵커가 이미지인 경우는 처리 대상에서 제외된다.

이를 위하여 우리는 다음과 같은 방법을 사용한다. 질의가 주어지면 기본모델에 기반하여 질의와 관련성이 있다고 간주되는 문서들을 구하여 이를 기본집합 A라 놓는다.

$A = \{ d | sim_0(d, q) > 0 \text{ for } all\ d\ in\ document\ collection\ \Omega \}$

그러나 식 (11)에 나타난 것처럼 질의에 대한 문서의 관련성은 앵커텍스트에 의해서도 생길 수 있다. 즉 어떤 문서의 입력링크(incoming link)의 앵커텍스트와 질의가 공통되는 색인어를 가진다면 이 문서의 RSV 값도 0 보다 크게 된다(왜냐하면 sim_2 값이 0보다 크기 때문이다). 그런데 우리는 색인과정에서의 색인어 추출 시에 문서 내의 앵커 텍스트 안에 있는 색인어도 색인어 추출 대상에 포함된다.

즉 그림 4와 같은 상황에서 질의가 "건강 보험"인 경우를 생각해 보자. 앵커텍스트 "의료 보험"도 문서 d_i 안의 텍스트의 일부이므로 문서 d_i의 색인어 리스트에 "보험"이 포함된다. 이 앵커텍스트는 문서 d_j를 가리키는 링크 1의 앵커텍스트로서 질의와의 공통 색인어 "보험"이 존재하므로 d_j의 sim_2 값이 0보다 클 수 있다. 즉 $sim_2(d_j) > 0$. 그런데 이것을 가능하게 한 앵커텍스트가 들어있는 d_i의 sim_0 값은 어떨까? "보험"이 d_i의 색인어로 추출되므로 분명히 sim0 (di) > 0. 결국 sim0 = 0인 문서는 분명히 다른 문서의 sim_2 값에 기여를 하지 못한다. 결론적으로 어떤 문서가 $sim_2 > 0$이려면 이 문서를 가리키는 링크를 포함한 문서는 반드시 집합 A에 속하는 문서일 것이다. 즉 어떤 문서로 하여금 sim_2 값을 가지게 하는 링크의 앵커텍스트를 포함하는 문서는 반드시 A에 속한다고 말할 수 있다. 그러나 sim_2 값은 있으나(0보다 크나) sim_0 값은 0인 문서도 존재할 수 있다.

$B = \{ d | sim_2(d, q) > 0 \text{ for } all\ d\ in\ document\ collection\ \Omega \}$

$E = \{ d | sim_0(d, q) + sim_2(d, q) > 0 \text{ for } all\ d\ indocument\ collection\ \Omega \}$

여기서 E = A∪B가 성립한다. F = A - B는 기본모델에 의한 (RSV에의) 기여는 있으나 앵커텍스트에 의한 기여는 없는 문서들을 나타낸다. D = B - A는 앵커텍스트에 의한 기여는 있으나 기본모델에 의한 기여는 없는 문서들을 나타낸다.

결국 우리는 효율적인 방법으로 E에 속하는 각 문서를 찾아야 하며 그것의 sim 값들을 계산하여야 한다. 기본이 되는 아이디어는 "sim_2를 가지는 문서는 반드시 A 에 속하는 문서에 있는 앵커텍스트로부터 기여를 받는다"는 사실을 이용하여 A 안의 각 문서가 가진 출력링크(outgoing link)의 앵커텍스트에 의한 기여가 가능하면 이 링크가 가리키는 문서에 이 기여를 제공하는 것이다.

(1) 먼저 전통적인 정보검색 기법에 의하여 A에 속하는 문서를 알아내고 문서마다 RSV 값으로 sim_0을 부여한다.

(2) A의 내용을 E 에 복사한다.

(3) A 내의 각 문서 d 마다 그 안에 포함된 출력링크(outgoing link)의 앵커텍스트 L 마다 질의 q와 L 사이에 공통색인어가 있나 본다. 있으면 다음 두 작업을 수행한다(L의 링크가 가리키는 문서가 d'이라 하자) :

  i) d'의 RSV에 L에 의한 점수를 더해 준다.

  ii) d'이 E 안에 아직 없다면 E에 넣어 준다.

위 방법은 문서집단 안의 모든 문서에 대하여 처리하지 않고 A 안에 문서만 처리하므로 시간이 많이 절약된다고 할 수 있다. 위 작업을 위해서는 전처리 단계에서 모든 문서마다 그 안에 있는 앵커텍스트 및 가리킴을 받는 문서가 무엇인지 조사하여 알고 있어야 한다.

5. 기타 기법

5.1 제목 정보의 이용

웹 문서는 항상 제목을 가지고 있으며 웹 문서의 검색에서는 제목이 매우 중요하다. 즉 제목에 나타난 색인어가 제목이 아닌 문장에 나온 색인어보다 훨씬 문서의 주제를 더 많이 암시한다. 이 점을 반영하기 위하여 우리는 어떤 색인어가 제목에 나타나면 색인어빈도수를 1 보다 큰 값 h 만큼 증가시킨다(현재 h = 5로 함). 이렇게 함으로써 제목에 나타난 색인어에 더 큰 중요성을 부여한다.

5.2 문장-질의 공통색인어수에 의한 계층화

지금까지 소개한 여러 정보를 이용하여 계산된 검색 기준값 RSV에 의하여 답으로 출력할 문서 및 그 순위가 결정된다. 그러나 우리는 이 정답으로 제시될 문서들의 순위화리스트(ranked list)에 최종의 계층화(stratification)에 의한 재순위화(re-ranking) 단계를 거친다. 여기서의 계층화의 기반은 문서 안의 문장과 질의의 공통 색인어 수(cic; common index-term count)이다. 문서는 여러 문장을 가지고 있으므로 질의와의 공통 색인어수가 가장 많은 문장이 이용된다. 문서마다 정해지는 cic는 다음과 같이 정의된다. 여기에서 s_i는 문서안의 각 문장(의 색인어 집합)을 나타내며, | |는 집합의 원소수를 나타낸다.

$$cic(d, q) = MAX_i\ |s_i \cap q| \quad \text{where i is the index for sentences in d.} \tag{12}$$

계층화의 핵심은 출력 대상이 되는 임의의 두 문서가 있을 때 RSV의 크기 여부에 관계없이 cic가 큰 문서가 작은 문서보다 앞 선 순위로 출력되어야 한다는 것이다. 직관적으로 말한다면 최종 검색 결과는 cic 값이 같은

문서들은 같은 계층에 속하며 먼저 *cic*에 의해 계층간의 순서가 결정되며 각 계층 안에서는 이에 속한 문서들이 *RSV*에 의하여 순서화된다는 것이다. 즉, *rank*(*d*)를 문서 *d*의 순위 번호라면 다음 조건이 만족되도록 순위를 다시 조정하는 것이다.[8)]

$$rank(d_i) \le rank(d_j) \text{ iff } cic(d_i, q) \ge cic(d_j, q) \text{ or}$$

$$[\ cic(d_i, q) = cic(d_j, q) \text{ and } RSV(d_i, q) \ge RSV(d_j, q)\ ]$$

5.3 문장 및 링크 정보에 의한 제한

지정페이지검색에서는 각 질의에 대한 정답은 질의와 관련성이 있는 많은 수의 문서가 아니고 질의가 원하는 소수의 문서이다. 그 이유는 질의가 매우 세부적으로 문서를 지정한다고 보기 때문이다. 따라서 재현율이 그리 중요하지 않으며 시스템도 질의가 원하는 그 문서를 찾는 것에 주력하여야 한다. 이러한 작업의 성격에 따라 우리는 어떤 문서가 문장-질의 유사도에 의한 기여나 링크정보에 의한 기여를 가지고 있지 않으면 검색결과에 포함시키지 않는다. 실험에서 이 기법은 크지는 않지만 약간의 성능 향상을 가져오는 것으로 나타났다.

6. 실험 및 성능 평가

실험에는 TREC을 위해서 NIST에서 제공한 웹문서 집단, 질의 집합, 각 질의에 대한 정답 문서를 포함하는 테스트 컬렉션을 이용하였다. 질의(토픽)의 총수는 150개이다. 지정페이지검색 작업에서는 질의마다 이 질의가 원하는 정보를 담고 있는 문서 하나 또는 두 개를 정답으로 한다. 문서들 사이에 순위는 없는 것으로 본다.

검색 작업의 특성 상 시스템 성능 평가의 기준으로는 기존의 정보검색에서 보통 사용하는 재현률(recall) 및 정확률(precision) 대신 최근에 TREC에서 채택하고 있는 MRR(mean reciprocal rank)를 이용한다. 이것은 각 질의마다 정답(answer) 문서가 시스템이 구한 검색결과 리스트에서 몇 번째 인지 가장 앞 선 순위를 구한 다음 이의 역수(reciprocal)를 구한다. 모든 질의에 대하여 이러한 역수를 구한 다음 이들의 평균(mean)을 구한 것이 MRR 이다. 가장 이상적인 시스템이라면 정답이 항상 검색결과에서 1위로 나오므로 MRR이 1.0이 된다. MRR이 0인 경우가 가장 열등한 시스템이 되는데 이는 모든 질의에 대하여 정답이 시스템이 내준 검색결과에 포함되지 않았음을 의미한다. 즉 모든 질의에 대해 전혀 정답을 찾지 못하였음을 나타낸다.

6.1 파라메타 값의 결정

RSV 값의 계산을 위해서는 파라메타 α, β, k 의 값이 정해져야 한다.[9)] 이는 여러 가능한 값에 대하여 실험을 수행하여 좋은 결과를 주는 값으로 결정한다.

• α의 결정

그림 5는 여러 가지 β에 대하여 α를 변화시키면서 MRR을 측정한 것이다. 이 때 k = 1로 놓고 실험을 하였다. 모든 β에 대하여 α가 1인 경우가 가장 높은 성능을 나타내고 있다. 따라서 α = 1로 결정된다.

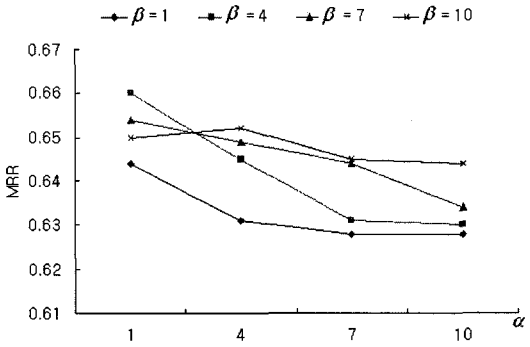

그림 5 α의 변화에 따른 성능 측정

• β의 결정

그림 6과 같이 여러 α에 대하여 β를 변화시키면서 MRR을 측정하였다. 이때 k = 1로 고정시켰다. 여기에서 알 수 있듯이 α에 따라서 가장 좋은 β 값이 다르게 나타나고 있다. 그러나 우리는 위에서 α를 1로 결정하였으므로 이 경우에 대하여 가장 좋은 β값을 선택하였다. 즉 β = 4로 결정하였다. 이 때의 성능은 다른 모든 α, β 조합의 경우보다 좋은 성능을 나타내므로 문제가 되지 않는 선택이다.

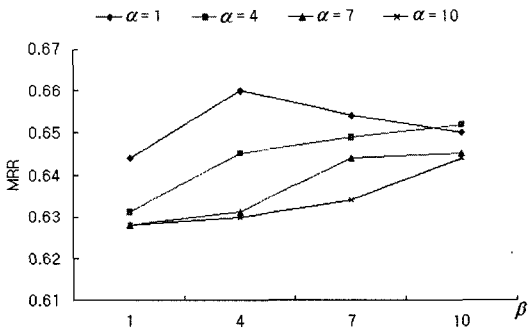

그림 6 β의 변화에 따른 성능 측정

• k 값의 결정

최적의 α = 1, β = 4 에 대하여 k 값의 변화에 대한 MRR 값의 측정 결과는 그림 7과 같다. 따라서 k = 5로 결정된다.

8) 순서화리스트에서 rank 값이 작을 수록 앞 선 순위이다.

9) 여기에서 α는 식 (7), β는 식 (11), k 는 식 (5)에서 사용된 파라메타들이다.

표 3 여러 기법에 따른 성능 측정[10)]

	기법의 결합	MRR	Top 10		Not found	
			Count	%	Count	%
1	sim_0	0.385	86	57.3	39	26.0
2	sim_0 + title	0.415	95	63.3	33	22.0
3	sim_0 + title + sim_1	0.567	117	78.0	22	14.7
4	sim_0 + title + sim_2	0.445	99	65.7	23	15.3
5	sim_0 + title + sim_1 + sim_{2a}	0.592	119	79.0	17	11.4
6	sim_0 + title + sim_1 + sim_{2b}	0.608	122	81.0	17	11.0
7	sim_0 + title + sim_1 + sim_2	0.623	122	81.3	14	9.3
8	sim_0 + title + sim_1 + sim_2 + cut	0.630	122	81.3	14	9.3
9	sim_0 + title + sim_1 + sim_2 + cut + str	0.698	128	85.3	13	8.7

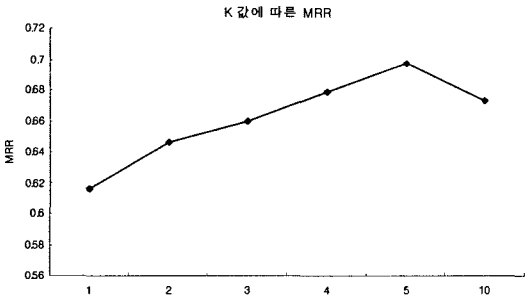

그림 7 k에 따른 성능 측정

6.2 성능 평가

표 3은 지금까지 소개한 여러 기법들의 결합에 따른 시스템의 성능을 나타낸다. "Top 10" 열은 전체 150개의 질의 중에서 시스템에 정답이 10위 이내에 포함된 질의의 수 및 비율을 나타낸다. "Not found" 열은 정답이 검색결과에 포함되지 못한 질의의 수 및 전체 질의에서의 비율을 나타낸다.

행 1은 기본모델 만을 사용한 시스템의 성능이다. 즉 식 (4)를 이용한 전통적인 정보검색 기법에 기반한 시스템에 의한 것이다. 그러나 5.1 절에서 설명한 제목 정보를 이용함으로써 행 2의 성능이 되었는데 이는 제목이 웹 검색에서 매우 중요한 역할을 함을 나타낸다.

행 3은 기본모델에 문장-질의 유사도를 사용한 경우이다. 이것으로 문장-질의 유사도에 의하여 많은 성능향상이 가능함을 알 수 있다. 행 4, 5, 6, 7는 링크 정보를 이용한 경우들에 대한 것인데 방법 1, 2 모두 성능향상에 기여함을 볼 수 있다. 기여의 정도는 방법 2가 방법 1 보다 더 좋은 기여를 하는 것으로 나타났다.

문장-질의 유사도나 링크에 의한 유사도 점수를 받지 못하는 문서를 제거한 경우가 행 8에 나타나 있다. 이 기법의 사용은 크지는 않지만 약간의 향상을 가져오고 있다. 행 9을 8과 비교하면 문장-질의 공통색인어 수에 의한 계층화에 의한 기여를 알 수 있다. 여기에서 놀라운 것은 이 기법이 문장-질의 유사도를 사용함으로써 얻는 기여와 거의 비슷한 성능 향상 효과가 있다는 것이다.

본 시스템의 성능과 다른 여러 연구 집단 시스템과의 성능을 비교한 것이 그림 8에 있다. 이에 의하면 TREC에 참가한 대부분의 시스템에 비하여 짙은 검은 색으로 표시된 본 시스템의 성능[11)]이 우수함을 나타낸다.

7. 결 론

본 논문에서는 웹 정보검색 시스템의 성능 향상을 위한 여러 기법에 관하여 소개하였다. 성능향상에의 가장 많은 기여는 제목 정보를 이용하는 기법이다. 이는 웹 정보검색에서 웹 페이지의 제목에 나타난 내용이 문서의 내용을 대변하는데 매우 유용하게 이용될 수 있다는 것을 의미한다. 두 번째로 문장과 질의 사이의 유사 정도에 관한 정보이다. 질의에 나타난 색인어가 문서의 동일 문장에 집중적으로 나타난 경우일수록 관련도가 높은 것으로 관측되었다. 그리고 웹 문서의 큰 특징 중의 하나인 링크정보의 이용도 기존의 연구 결과와 달리 성능향상에 유용하게 이용될 수 있음이 밝혀졌다. 특히 입력링크의 앵커텍스트가 이 링크가 가리키는 문서의 내용과 많은 관계를 가지는 것으로 판명되었다. 본 연구의 실험에서 밝혀진 독특한 점은 문장-질의 공통색인어 수에 의한 계층화도 성능 향상에 많은 기여를 한다는 점이다. 본 연구는 여러 가지 다양한 기법을 이용함으로써 웹 정보검색의 성능향상을 도모할 수 있음을 보여 주었다.

문장-질의 유사도의 계산은 큰 오버헤드를 야기한다. 그러나 분석 결과 전체 문서량의 증가에 따른 오버헤드의 증가는 선형적인 것으로서 기존시스템의 경우와 다

10) 이 표에서 cut은 5.3 절의 문장 및 링크 정보에 의한 제한 기법을 나타내며 str은 5.2 절의 계층화 기법을 나타낸다.

11) 이것은 표 3의 sim_0 + title +sim_1 +sim_2 +cut +str이 나타내는 시스템의 성능을 말한다.

른 특별한 대책을 필요로 하지는 않는 것으로 밝혀졌다. 따라서 실용적인 시스템에서의 사용도 가능한 것으로 보인다.

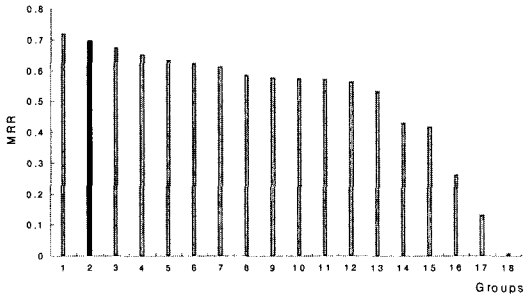

그림 8 여러 연구집단의 시스템 성능

참 고 문 헌

[1] D. Harman, "The TREC Conferences," In Readings in Information Retrieval, pp. 247-256, Morgan Kaufman, 1997.

[2] E. Voorhees and D. Harman, "Overview of TREC 2001," Proc. of the Tenth Text Retrieval Conference TREC 2001, May, 2002.

[3] Sumio Fujita, "More reflections on 'aboutness' TREC-2001 evaluation experiments at Justsystem," Proc. of the Tenth Text Retrieval Conference TREC 2001, May, 2002.

[4] N. Craswell and D. Hawking, "Overview of the TREC-2002 Web Track," Proc. of the Eleventh Text Retrieval Conference TREC-2002, NIST, May, 2003.

[5] National Institute of Informatics, "NTCIR Workshop 3 Meeting OVERVIEW," Working Notes of the Third NTCIR Workshop Meeting, October 8-10, 2002.

[6] P. Bailey, N. Craswell and D. Hawking, "Engineering a multi-purpose test collection for Web retrieval experiments," Technical report, CSIRO, 2001.

[7] E. Voorhees, "Variations in relevance judgements and the measurement of retrieval effectiveness," Information Processing and Management, 36, pp. 697-716, 2000.

[8] G. Salton, *Automatic Text Processing*, Addison-wesley, 1989.

[9] G. Salton, A. Wong, and C. S. Tang, "A Vector Space Model for Automatic Indexing," *Communications of the ACM*, 18:11, pp. 614-620, Nov, 1975.

[10] J. Perez-Carballo and T. Strzalkowski, "Natural language information retrieval: progress report," Information Processing and Management, Vol. 36, pp.155-178, 2000.

[11] J. Kleinberg, "Authoritative sources in a hyerlinked environment," Technical Report RJ 10076, IBM, 1997.

[12] D. Hawking, "Overview of the TREC-9 Web Track," Proc. of the Ninth Text Retrieval Conference TREC 2000, NIST, May, 2001.

[13] J-M Lim, H-J Oh, S-H Maeng and M-H Lee, "Improving efficiency with document category information in Link-based retrieval," In Proc. of the Information Retrieval on Asian Languages Conference, 1999.

박 의 규

1995년 연세대학교 전산학과(이학사). 2001년 연세대학교 전산학과(이학석사). 2001년~현재 연세대학교 전산학과 박사과정 1995년~1999년 (주)리딩텍 근무. 관심분야는 정보검색, 구문분석, 형태소분석

나 동 열

1978년 서울대학교 전자공학과(공학사) 1980년 한국과학기술원 전산학과(이학석사). 1989년 미시간주립대학교 전산학과(공학박사). 1980년~1990년 한국전자통신연구원 선임연구원. 1991년~현재 연세대학교 전산학과 교수. 관심분야는 자연어처리, 정보검색, 인공지능

장 명 길

1988년 부산대학교 계산통계학과(이학사). 1990년 부산대학교 계산통계학과(이학석사). 2002년 충남대학교 컴퓨터과학과(이학박사). 1990년~1998년 5월 시스템공학연구소 선임연구원. 1998년 6월~현재 한국전자통신연구원 지식마이닝연구팀 팀장. 관심분야는 정보검색, 질의응답, 자연어처리