이질적 검색기와 컬렉션으로부터 검색된 복수 문서 리스트의 효율적인 융합 방법

이민호, 맹성현
충남대학교 컴퓨터과학과
e-mail: mhlee@irsun.chungnam.ac.kr

A Deterministic Fusion Method for Multiple Lists of Documents from Heterogeneous Search Engines

Min-Ho Lee, Sung-Hyon Myaeng
Dept. of Computer Science, Chungnam National University

요 약

본 논문은 분산, 독립적인 다수의 문서 컬렉션으로부터의 검색결과를 병합하는 컬렉션 융합(collection fusion)문제에 대한 효과적인 랭킹방법을 제시한다. 일반적인 컬렉션 융합 문제란 분산되어 있는 다수의 문서 컬렉션에서 독립적이고 능동적인 검색기들의 검색결과를 효과적으로 랭킹(ranking) · 병합하는 것인데, 각기 다른 특성을 가진 다수의 컬렉션을 동일한 검색기를 통하여 검색된 결과를 병합하는 환경과 서로 다른 알고리즘을 갖는 검색기를 통한 검색 결과 병합 환경으로 나누어 질 수 있다. 본 논문에서는 서로 다른 특성을 갖는 다수의 컬렉션을 서로 다른 알고리즘을 갖는 검색기들을 통하여 검색한 결과를 병합하는 방법을 제시한다. 각 컬렉션에 학습 질의를 넣어 얻은 정보를 토대로, 실제 질의를 넣었을 때 각각의 컬렉션에서 나온 결과가 통합 결과 집합에서 차지하는 비율과 각 문서의 순위를 결정한다. 기존 연구에서 사용한 방법들은 랜덤성에 의존한 비결정적인 랭킹 방법을 제시하거나, 단순하게 검색결과 집합의 문서 수를 바탕으로 인터리빙(interleaving)하는 방법을 제시하였다. 본 논문에서는 학습 질의에서 나온 정보를 기반으로 결정적이면서도 보다 효과적인 랭킹 방법을 제시한다.

1. 서론

네트워크, 특히 인터넷의 발달은 검색 엔진이 가장 유용하고도 꼭 필요한 자원이 되도록 만들었다. 하지만, 인터넷상에서 정보를 찾는 것은 검색엔진이 어디에 있는가, 검색엔진을 어떻게 사용하여야 하는가, 어느 검색엔진을 사용하여야 자신이 원하는 정보를 잘 찾을 수 있는가를 알아야 하는 새로운 문제를 대두시켰다.[1] 이러한 문제점을 해결하기 위하여 사용자가 원하는 정보가 있을 만한 여러 개의 검색엔진을 선택, 질의 후 얻은 결과를 통합하여 사용자에게 보여주는 메타 검색엔진이 등장하였다. 이를 구현하기 위해 필요한 중요한 과정이 정보 요구에 맞는 검색엔진을 선택하는 것과 검색되어 나온 결과들을 효과적으로 통합하고 문서의 순위를 결정(ranking)하는 부분이라 할 수 있다. 특히 문서순위 결정방법은 높은 순위를 갖는 문서일수록 질의에 대한 만족도가 크며, 사용자는 높은 순위의 문서를 우선적으로 검토함으로써 필요한 정보를 얻는데 소모되는 시간을 최소화할 수 있으므로 무척 중요하다.

검색 결과를 효과적으로 통합하고 순위를 결정하는 문제를 일반적으로 컬렉션 융합문제라고 하는데, 이는 크게 두 가지로 생각할 수 있다.

첫째로, 각기 다른 특성을 가진 다수의 컬렉션을 동일한 검색기를 통하여 검색된 결과를 융합하는 환경과, 둘째로 서로 다른 알고리즘을 갖는 검색기를 통한 검색결과 병합 환경으로 나누어 볼 수 있다. 본 논문에서는 후자에 초점을 맞추어 서로 다른 탐색 알고리즘을 갖는 검색기를 통하여 검색한

결과를 효과적으로 병합하고 순위를 결정하는 방법을 제시한다. 각 컬렉션에 학습질의를 통해 얻은 문서 결과 정보를 토대로, 실제 질의를 넣었을 때 각각의 컬렉션에서 나온 결과가 통합 결과 집합에서 차지하는 비율과 각 문서의 순위를 결정한다. 랭킹 방법에 대한 기존 연구에서 사용한 방법들은 랜덤성에 의존한 비결정적인 문서순위 결정방법을 제시하거나, 단순하게 검색결과 집합의 문서 수를 바탕으로 인터리빙(interleaving)하는 방법을 제시하였다. 본 논문에서는 학습질의를 통한 검색 결과를 기반으로 결정적이면서도 보다 효과적인 방법을 제시한다.

다음절에서는 컬렉션 융합문제를 정의하고, 3 절에서는 기존 연구에서 사용한 방법에 대한 설명 및 문제점들을 지적한다. 4 절에서는 본 논문에서 제안한 방법에 대하여 설명하고 , 5 절에서는 실험 및 결과를 보여주며, 끝으로 6 절에서 결론을 맺는다.

2. 컬렉션 융합 문제

컬렉션 융합문제는 Voorhees 의 논문에 잘 정의되어있다.[2] 서로 다른 특성을 갖는 컬렉션을 하나씩 가지고 있는 C 개의 검색기 I 들이 있다고 가정하자. 이때 각 검색 엔진들은 서로 다른 알고리즘으로 문서들의 순위를 정할 수 있으며 주어진 질의 Q 에 대하여 각 컬렉션들은 n_i 개의 적합문서를 가지고 있다고 하자. 사용자가 전체 컬렉션에서 검색하고자 하는 문서의 총 수를 S 라고 하며 각 검색 엔진들로부터 가져오는 문서의 수를 s_i 라하면,

$$\bigcup_{i=1}^{C} si = S$$

이고,

$$\max(\sum_{i=1}^{C} F(s_i))$$

인 s_i를 찾을 때 가장 우수한 검색성능을 보이게 된다. ($F(s_i)$는 각 검색 결과 문서 리스트에서의 적합 문서 수이다.) 하지만, 이 $F(s_i)$값은 알 수 없기 때문에 추정하여 n_i 가 가장 큰 컬렉션 i 에서 더 많은 문서를 추출하여야 한다. 또한, 사용자는 높은 순위에 위치한 문서들부터 검토하려 하기 때문에 적합한 문서들이 높은 순위에 위치할수록 성능이 좋은 검색 엔진이라 할 수 있다. 각 검색 엔진들이 검색한 문서 리스트들에서 추출할 문서 수를 정하고 융합하는 방법에는 다음과 같은 것들이 있다.

- 각 컬렉션은 다른 컬렉션들과 같은 수의 적합 문서를 포함하고 있다고 가정하여 각 컬렉션에서 같은 수의 문서들을 추출하고 문서들의 순위는 각 컬렉션에서 순번대로 할당(interleaving)하여 정하는 방법이 있다. 서로 다른 컬렉션은 서로 다른 특성을 가지고 있어서, 적합문서의 수도 같지 않기 때문에 이 방법은 매우 낮은 성능을 얻게 된다. 이 방법을 사용하여 검색한 결과와 각 컬렉션들을 합쳐 하나의 컬렉션으로 만들어 검색한 결과를 비교 하였을 경우 40%의 성능 감소를 보인다. [3]

- 각 검색기가 계산한 문서와 질의간의 유사도(similarity)값을 비교 가능하다고 가정하고 모든 검색된 문서들 중 가장 유사도가 큰 S 개의 문서들을 선택하는 방법이 있다. 하지만, 문서에는 컬렉션에 의존적인 빈도수 또는 가중치가 들어있고 각 검색 엔진들마다 문서와 질의와의 유사도를 계산하는 방식이 다양하기때문에 위와 같은 방식은 옳다고 할 수 없다. 이 방법이 옳지 않다는 것을 Dumais [3]가 실험을 통하여 증명하였다.

- 학습질의의 결과로부터 적합문서가 어느 컬렉션에 많이 있는가를 추정하는 방법이 연구되었다. 학습질의를 통하여 각 컬렉션의 내용과 검색 패턴을 모델링한 후 새로운 질의에 대해서는 학습질의와의 유사정도를 계산하여 각 컬렉션에서 검색하여야 할 문서의 수를 계산한다.

3. 기존 연구

학습질의를 통한 적합문서의 분포를 추정하는 방법에 관한 연구는 Voorhees[2][3][5]가 주로 수행하였는데, 그는 새로 입력된 질의와 유사한 k 개의 학습 질의의 검색이력을 사용한다. 즉, 각 질의가 검색한 적합 문서의 수를 바탕으로 실제 질의를 넣었을 때 각 문서 컬렉션에서 추출되어야 하는 문서 수와 전체 순위를 정하는 적합문서 분포 모델링 방법(Modeling Relevant Document Distributions)을 고안하였다. 다시말하면, 학습질의를 통하여 각 컬렉션으로부터 검색된 결과 집합을 대상으로 적합성 판단을 하여 적합 문서가 각 컬렉션에 얼만큼 많이 있는지를 나타내는 분포 모델을 생성하고, 이를 이용하여 각 컬렉션에서 추출할 문서의 수를 구하는 방법이다.

실제 질의 q 에 대한 적합 문서 분포는 질의 q 와 가장 유사한 k 개의 학습질의의 적합문서 분포로부터 구한다. 즉, 유사한 k 개의 학습질의에 대한 각

컬렉션에서의 학습 적합문서의 수를 합한 결과가 최대가 되게 각 컬렉션에서 추출할 문서의 수를 선택하는 것이다. 현재 질의에 대한 각 컬렉션에서의 적합 문서 분포가 일단 구해진 후에는, 이 분포 정보와 검색되어야 할 전체 문서의 수 S가 최대값 계산과정으로 전달되어 각 컬렉션에서 검색된 문서 중 적합 문서 수를 최대값으로 하는 S_i를 찾아내게 된다. 이에 따라 계산된 값 S_i는 각 컬렉션에서 추출할 검색 문서의 수가 된다.

각 컬렉션으로부터 랭킹된 검색 결과 집합은 다시 전체적인 랭킹작업을 거치게 된다. 전체 순위 중 r번째 순위에 해당하는 문서를 선택하기 위해서는, 문서의 수에 편중된 성향을 보이는 C-면 주사위(C-faced die)라는 방법을 사용한다. C-faced die는 C개의 컬렉션이 있을 때, 서로 다른 확률을 갖는 C면의 주사위를 던져 문서를 선택하는 방법을 말한다. 전체 순위 r 다음에 위치할 문서는 앞 순위에서 선택된 문서를 제외한 나머지 검색 결과 집합들을 대상으로 다시 결정하게 된다. 컬렉션 i에서의 검색 결과에서 다음 문서를 취할 확률은 다음과 같다.

$$P_i = \frac{g_i}{\sum_{j=1}^{C} g_j}$$

g_i는 상위 r까지의 문서를 선택하고 각 컬렉션 i에서 검색된 결과 리스트에서 남은 문서들의 수이다.

이때, 각 컬렉션 I_i들간의 추출 확률 간격은

$$T_i = [a_i, b_i)$$

이며

$$a_1 = 0$$
$$a_i = b_i - 1, \qquad i = 2,\ldots,c$$
$$b_i = a_i + p_i, \qquad i = 1,\ldots,c$$

이다. 무작위 수 ran ∈ [0,1]을 얻어, ran ∈ Ti 인 검색 리스트 i에서 문서를 추출하는 것이다.

위의 방법은 랭킹단계에서 무작위 추출(random)에 의존하는 알고리즘을 사용하기 때문에 같은 질의를 하더라도 문서의 순위가 매번 달라지는 비결정적인 방법이라는 문제가 있다.

Yager는 Voorhees 방법의 이러한 문제점을 제기하고 랜덤에 의존하지 않는 결정적인 방법을 제시하였다.[6] Voorhees의 방법은 '각 컬렉션에서 검색되어진 문서 리스트 중 문서의 개수가 가장 많은 리스트가 가장 적합한 문서를 가지고 있을 확률이 높다' 라는 가정을 가지고 있으며 문서 순위 결정 방식에서 검색 문서 리스트의 길이가 긴 컬렉션에서 더 높은 확률을 가지고 문서를 선택하게 된다. Yager는 이 가정과 기본적인 내용은 같지만 랜덤성을 배제하여 결정적인 랭킹 방식을 제안하였다. Yager는 '각 컬렉션에서 검색되어진 문서리스트 중 문서의 개수가 가장 많은 리스트가 가장 적합한 문서를 가지고 있다.' 라는 가정을 한다. 즉, 각 컬렉션에서의 검색 결과 리스트가 가장 긴 리스트에서 문서를 먼저 추출한다. 각 컬렉션이 갖는 선택지수는 다음과 같다.

$$V_i = \frac{g_j}{\sum_{j=1}^{C} g_j}$$

위와 마찬가지로, g_i는 각 검색된 결과 리스트에서 현재까지 추출되고 남은 문서들의 수이다. Voorhees와의 차이는 확률 개념을 제거하여 결정적으로 컬렉션을 선택하는 것이다. 즉, Vi가 가장 큰 수를 갖는 컬렉션 i의 검색 결과에서 문서를 선택한다.

위 두 방법 모두 각 컬렉션에서 추출하여야 할 문서의 개수를 정하는 최대값 계산 과정을 사용하는데, 최대값을 찾을 때까지 각 검색결과 리스트에서 문서의 수를 하나씩 늘려 나가는 완전 탐색(exhaustive search) 방법을 사용하므로 컬렉션 수가 많아지거나 학습질의 검색 결과가 많을 경우 상당한 계산시간을 요하게 된다. 완전탐색을 하는 경우 $O(S^c)$ (C는 컬렉션 개수이고, S는 최종 검색 문서 수) 의 계산을 필요로 한다.

4. 제안방법

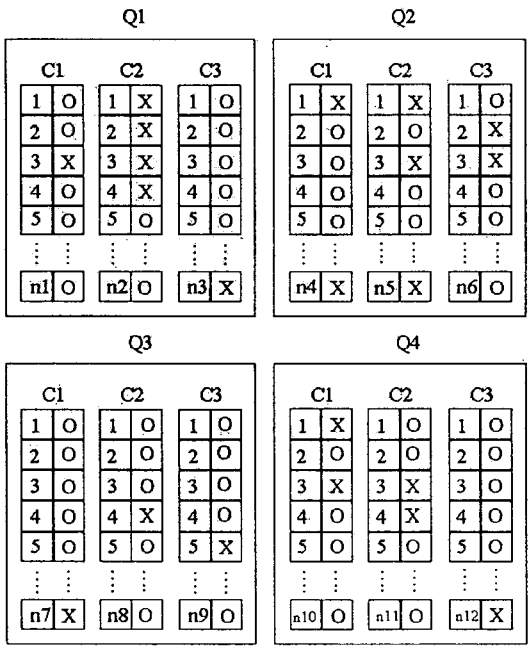
3절에서 살펴보았듯이 Voorhees의 방법은 랜덤성에 의존하기 때문에 같은 질의를 넣더라도 검색 결과가 매번 달라지는 문제점을 가지고 있으며, Yager의 방법은 검색 결과 리스트의 길이에 의한 인터리빙 방법을 사용한다. 본 논문에서는, 결정적이며 Yager의 방법에 비해 보다 효과적인 랭킹 방법을 제시한다.

- RRJ1 (Ranking with Relevance Judgement 1)

본 논문에서 제시하는 컬렉션 융합 과정의 기본적인 방법은 Voorhees의 융합 방법과 비슷하지만 각

컬렉션에서 추출하여야 할 문서 수를 미리 계산하는 과정이 없고 문서 순위 결정방식에 있어서도 차이를 보인다. 학습 단계에서는 각 질의를 질의에서의 단어 빈도수를 사용한 질의 벡터를 생성한다. 이때, 학습질의를 각 컬렉션에 넣어 검색한 결과를 적합성 판단하여 적합 문서 판단 정보파일에 저장한다. 적합성 판단 정보는 학습 질의마다 각 컬렉션에서 검색된 문서들을 순위별로 적합성 여부를 판단하여 기록하여 놓은 것이다. [그림 1]은 적합 문서 판단 정보의 구조를 보여주고 있다.

전체 순위 r에 해당하는 문서를 선택할 때에는 학습단계에서 얻어진 적합 문서 판단 정보에서 r번째 순위에 해당하는 문서들 중 적합한 문서를 갖는 질의의 수가 가장 많은 컬렉션을 선택하여 그 컬렉션 검색 리스트에서 문서를 가져온다. 만약, 두 개 이상의 컬렉션에서 적합 문서 판단 값이 같은 경우에는 바로 위 순위 r-1에서 선택한 컬렉션에서 선택하게 된다. 이것은 연속적으로 적합 문서를 생성하는 컬렉션에 다음 순위를 위한 적합 문서들이 모여 있을 가능성이 높다는 가정에 의한 것이다.



[그림 1] 적합 문서 판단정보 구조

p개의 학습 질의를 C개의 서브 컬렉션 $I_1, I_2, \ldots I_c$에서 학습한 상태이고, 실제 질의와 유사한 k개의 학습질의가 $Q_1, Q_2, \ldots Q_k$ 으로 선택되었다고 하자. 전체 순위 r에 위치할 문서는 다음 식에 의해 V_i 값이 가장 큰 컬렉션 I_i에서 선택하게 된다.

$$V_i^r = \sum_{j=1}^{k} R(Dr, Q_j)$$

i= 1,2,...,c

R(Dr,Qj)는 i번째 컬렉션에서 Q_j로 질의하였을때, 순위 r번째로 검색된 문서 Dr의 적합 여부 함수로 적합한 문서일 때의 값은 1이고, 비적합 문서일 때의 값은 0이다.

예를 들면, 서로 다른 컬렉션 3개가 있다고 하고, 4개의 학습질의를 하였다고 하자. 학습질의 검색 결과 나온 문서들의 적합성 판단 정보가 [그림-1]과 같다고 하고 유사한 3개의 질의가 Q1, Q3, Q4라고 하였을 때, 순위 결정은 다음과 같이 한다.

1번째 순위에 해당되는 문서는 컬렉션 1에서의 순위 1번의 적합문서 수에 해당하는 V_1^1=(R(1,Q1) + R(1,Q3) + R(1,Q4)) 과 컬렉션 2에서의 순위 1번에 해당되는 적합문서 수 V_2^1 = (R(1,Q1) + R(1,Q3) + R(1,Q4)) 과 컬렉션 3에서의 순위 1번째 적합문서 수를 더한 값인 V_3^1 = (R(1,Q1) + R(1,Q3) + R(1,Q4)) 의 합을 비교한다. 여기에서는 2, 2, 3의 적합문서 합이 나왔으므로 컬렉션 3에서 문서를 선택하게 된다. 전체 순위 2번에 해당하는 문서 역시 같은 방법으로 V_1^2 = (R(2,Q1) + R(2,Q3) + R(2,Q4)) , V_2^2 = (R(2,Q1) + R(2,Q3) + R(2,Q4)) , V_3^2 = (R(2,Q1) + R(2,Q3) + R(2,Q4)) 중에서 큰 값을 선택한다. 3,2,3으로 가장 큰 값을 갖는 컬렉션이 둘 이상 나왔으므로 바로 위 순위인 1번째 순위에서 선택한 컬렉션 3에서 2위에 해당하는 문서를 선택하게 된다.

- RRJ2 (Ranking with Relevance Judgement 2)

순위 r을 정할 때 r보다 상위에 선정된 적합 문서 수를 합한 누적 적합 문서 수로 순위 결정하는 방법이다. 이것은 적합문서가 고루 퍼져있으나 각 순위에서 유사질의에 따른 적합문서의 합이 다른 컬렉션보다 작은 경우, 그 컬렉션에서 잘 선택이 되지 않는 경우를 해결하기 위한 것이다. [그림 2]는 각 순위에 적합한 문서를 검색할 질의의 수를 보여주는데, 이 경우 1번 컬렉션은 전체적으로 적합 문서가 고루 퍼져있다. 2번, 3번의 경우 특정 순위에서 적합 문서가 많은데 이런 경우 1번 컬렉션에서는 전혀 문서가 선택되지 않게 된다.

C1	C2	C3
2	3	0
1	2	1
1	0	2
2	0	3
2	1	1
2	0	3
1	2	0
2	2	0
2	2	0
1	0	2
1	0	2
2	0	2
1	2	0
1	1	0

[그림 2] C1 의 검색 리스트에 적합문서가 고루 분포되어 있는 경우

C1	C2	C3
3	0	0
3	0	0
3	0	2
3	0	3
2	0	1
2	0	3
1	0	3
0	0	3
0	0	3
0	2	2
0	1	1
0	3	1
0	3	0
0	3	0

[그림 3] C1 은 상위에, C2 는 하위에 적합문서가 집중적으로 분포되어 있는 경우

- RRJ3 (Ranking with Relevance Judgement 3)

상위에 선정된 적합 문서 수를 합한 값, 즉 RRJ2 에서 계산된 값을 Voorhees 의 방법에 의해 각 컬렉션에서 추출하여야 할 값으로 나누는 방법이다. 이것은 전체적으로는 적합 문서 수가 적지만 하위부분이 적합문서가 다른 집합에 비하여 특히 많이 몰려 있을 컬렉션이 하위 순위에 적합문서가 거의 없는 다른 컬렉션보다 선택이 되지 않는 경우를 방지하기 위하여 고안한 것이다. [그림 3]의 경우, RRJ2 를 사용하게 되면 상위 순위에는 1 번 컬렉션에서 중반 이후에는 계속 3 번 컬렉션에서 선택되게 되고, 2 번 컬렉션에서는 전혀 선택되지 않는 경우가 생긴다. 이때 2 번 컬렉션의 하위에 존재하는 많은 적합문서를 선택하기 위하여, 각 컬렉션에서 추출하여야 할 문서 수를 정하고 이 값으로 누적 적합문서 수를 나누면 어느 정도 2 번 문서의 하위 적합 문서들을 추출할 수 있다. 하지만, 이것은 Voorhees 의 최대값 계산과정을 이용하여야 하기 때문에 속도가 느리다는 단점을 갖는다.

위에서 설명한 방법들은 같은 질의에 대해서 항상 동일한 검색 결과를 내어놓는 결정적 방법이 되며 세 번째 방법을 제외하고는 각 순위에 해당하는 문서를 추출할 컬렉션을 결정하기만 하면 되므로 각 컬렉션에서 추출하여야 할 문서의 수를 정하기 위해 많은 계산을 할 필요가 없다. 그러므로 융합시간의 대부분을 차지하는 완전탐색 방식인 최대값 계산을 하지 않아 검색 시간을 단축시킬 수 있다.

5. 실험결과

실험 데이터는 TREC-3 collection [7]중 디스크 1 번을 대상으로 하여 AP, DOE, FR,WSJ, ZIFF 다섯 개의 서브 컬렉션으로 전체 컬렉션을 나눈 후, TREC 의 질의 중 1-150 번 까지를 학습질의로 사용하였다. 실험질의를 사용하여 각 서브 컬렉션을 대상으로 검색한 후, 그 결과 중 상위 순위 200 개를 뽑아 적합문서 분포 정보를 생성하였다.

Yager 의 방법은 논문[5]에서 제시한 세가지 방법을 사용하고 각각을 Yager 1, Yager2, Yager3 이라 칭하였다. Yager 1 은 각 컬렉션에서 뽑아야 할 문서의 수에서 현재까지 각 컬렉션에서 추출하여 최종 사용한 문서의 수를 뺀 나머지가 가장 큰 컬렉션에서 문서를 상위랭크에 추출하는 방식이며, Yager2 는 반대로 추출된 문서가 적은 컬렉션 리스트에서 추출하는 방식이다. Yager3 은 뽑아야 할 문서의 수에서 현재까지 추출한 문서의 수를 뺀 나머지를 2 로 나눈 값이 가장 큰 컬렉션에서 추출한다.

C-faced die 방법과 Yager 의 방법들, RRJ3 방식은 각 컬렉션에서 추출해 내어야 하는 문서 수를 정하는 과정이 최대값 프로시저를 사용하므로 시간이 오래 걸리기 때문에 더 많은 학습 문서를 사용하지 못 하였다.

검색기로는 벡터 검색방식을 사용하는 코넬 대학에서 개발된 SMART 시스템[8]을 사용하였으며 가중치 기법은 코사인 정규화를 사용하는 가중치 기

법, ntc, ntc [1]를 적용하였다. [9]

테스트 질의는 topic 151-200 번까지 50 개를 사용하였으며 검색되어야 하는 전체 문서 수 S 는 200 으로 하였다. 각 방식을 적용하여 컬렉션 융합을 한 후 결정된 200 개의 문서 중 10, 20, 30, 100 위까지의 정확도를 비교해본 결과, RRJ 1 방법이 전반적으로 우수한 결과를 보였으며, Voorhees 의 C-face die 방법은 확률로써 순위를 결정하는 방식임에도 불구하고 Yager 의 방법에 비해 우수하였다.
C-faced die 방법은 비결정적인 방법이므로 향상된 증가치 계산에는 결정적인 방법인 Yager1 을 사용하였다. 상위 30 위안에 들어가는 문서의 정확도에 있어서 기본 비교 대상인 Yager1 에 비해 RMRJ1 의 방법은 25.5%의 성능향상을 보였다. ([표 1] 참조)

이 실험 결과는 기존 방법들이 RRJ1 방법보다 적합한 문서들을 상위에 위치시키지 못함을 의미한다. RRJ1 이 우수한 성능을 보인 원인은 RRJ1 과 RRJ2 를 제외한 나머지 방식들이 각 컬렉션에서 뽑아야 할 개수를 전체 200 개의 문서를 추출하기 위한 S 값에서 계산하기 때문이다. 즉, 200 위까지의 문서 전체를 대상으로 한 정확도는 모두 같지만 각 순위에 위치한 문서가 실제 적합한 것인가에 해당되는 정보는 전혀 없는 상황에서 순위결정을 하기 때문이다. 실제로 상위 순위 20, 30, 100 위 안에 들어가는 문서들의 정확도는 하위 순위까지 많이 선택하여 정확도를 비교할수록 모든 방법들이 거의 비슷해짐을 알 수 있다.

사용자는 검색기의 결과 중 상위 순위에 위치하는 문서를 우선 보고 싶어하며, 높은 순위의 문서를 우선적으로 검토함으로써 필요한 정보를 얻는데 소모되는 시간을 최소화할 수 있다. 그러나, 사용자는 자신이 찾고자 하는 정보를 충분히 검색하기 위하여 전체 검색 문서 S 를 크게 잡거나, 사용자가 어느 순위까지의 문서를 얻고자 할지를 검색기 입장에서는 알 수 없기 때문에 검색기는 충 검색하여야 할 문서 수를 크게 잡을 수 밖에 없다.
이렇게 총 검색 문서 수를 크게 잡으면, 전체 정확도는 RRJ1 와 RRJ2 를 제외한 모든 방법이 같게 되지만 상위 순위 문서들의 정확도는 상대적으로 떨어지게 된다.

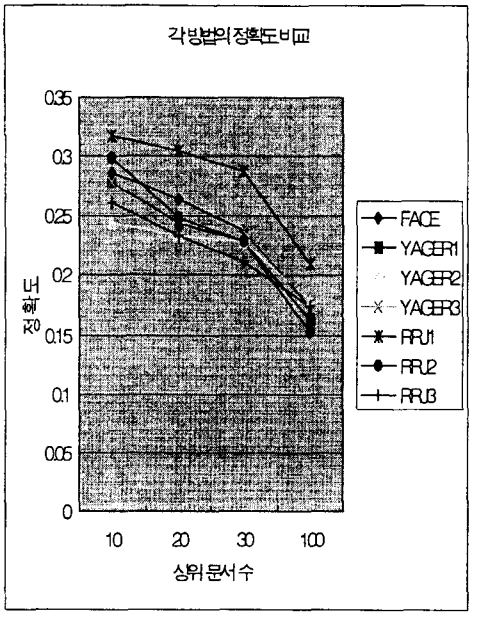
RRJ2 와 RRJ3 는 기존 방식과 거의 유사한 성능을 보였지만 기대에는 못 미쳤다. RRJ2 방법의 경우는 TREC 컬렉션이 특성별로 잘 분류가 되어 있어 한 컬렉션이 적합한 문서수가 월등히 많아, 적합문서 수를 누적하는 방식으로는 다른 소수의 적합한 문서를 갖는 경우가 선택되는 일이 거의 없었다.

향후 연구로는 현재의 실험 환경을 확장하여 보다 대규모 컬렉션 및 학습질의가 검색에 어떠한 영향을 주는지 알아보는 것이 필요하다. 또한, 현재의 실험환경에서는 컬렉션을 문서 종류별로 분류하여 구축하였는데, 각 컬렉션이 다양한 종류의 문서로 구성되었을 때 보이는 현상도 연구되어야 한다. 마지막으로 각 알고리즘이 갖는 장점들을 혼합하는 방법도 고려되어야 한다.

순위 / 방법	10	20	30	100
Yager 1	0.2780	0.2440	0.2293	0.1628
Yager 2	0.2440 (-12.2%)	0.2340 (-4.0%)	0.2280 (-0.5%)	0.1782 (9.4%)
Yager 3	0.2780 (0%)	0.2490 (2.0%)	0.2373 (3.4%)	0.1764 (8.3%)
C-faced Die	0.2860 (2.8%)	0.2640 (8.1%)	0.2387 (4.0%)	0.1740 (6.8%)
RRJ 1	0.3180 (14.3%)	0.3050 (25%)	0.2880 (25.5%)	0.2094 (28.6%)
RRJ 2	0.2980 (7.1%)	0.2490 (2.0%)	0.2293 (0%)	0.1520 (-6.6%)
RRJ 3	0.2651 (-5.7%)	0.2330 (-4.5%)	0.2112 (-7.8%)	0.1750 (7.4%)

[표 1] 상위 순위로 결정된 30, 50, 100, 150 개의 문서의 정확도 비교. ( )안의 숫자는 Yager1 의 방법을 사용한 순위 결정 정확도를 100 으로 보았을 때와 비교한 상대적 정확도 차이.

[1] 가중치 계산 방식에 필요한 요소 중 단어 빈도요소는 tf, 문서 빈도 요소는 ln N/n, 정규화 요소는 코사인 정규화를 사용하는 경우임.



[그림 4] 각 방법의 정확도를 나타낸 그래프

6. 결론

인터넷의 발달로 인해 등장한 메타 검색엔진을 효과적으로 구현하기 위해서는 사용자가 원하는 정보 요구를 잘 검색해 줄 수 있는 검색 엔진의 선택 기술과 검색 결과의 병합, 순위를 결정하는 기술이 꼭 필요하다. 검색 결과를 효과적으로 통합, 순위를 결정하는 문제를 일반적으로 컬렉션 융합문제라고 하는데, 이는 각기 다른 특성을 가진 다수의 컬렉션을 동일한 검색기를 통하여 검색된 결과를 병합하는 환경과 서로 다른 알고리즘을 갖는 검색기를 통한 검색결과 병합 환경으로 나누어 질 수 있다. 다수의 컬렉션을 이질적인 검색기를 통하여 검색한 결과를 병합, 순위 결정하는 방법에 있어서 기존의 방법들은 랜덤성에 의존하여 같은 질의에 대해 다른 검색 결과를 내어놓는 비결정적인 랭킹방법을 제시하거나, 검색 결과 문서 수를 바탕으로 단순하게 인터리빙하는 방법을 제시하였다. 본 논문에서는 학습 질의에서 나온 적합 문서 판단 정보를 기반으로 계산 복잡도가 매우 낮은 장점을 가지면서 보다 우수한 검색 신뢰도를 보여주었다. 또한 융합 알고리즘이 결정적이므로 일관성을 가진 결과를 생성한다. 이를 통해 메타 검색엔진의 성능을 향상시킬 수 있을 것이며, 앞으로 대규모 컬렉션에서의 성능 평가와 학습되지 않은 순위에서의 순위 결정 방법에 대한 연구가 필요하다.

7. 참고문헌

[1] Robert R. Korfhage. "Information Storage and Retrieval", Wiley Computer Publishing, 1997.

[2] Ellen M. Voorhees, Narenda K. Gupta, and Ben Johnson-Laird. "The Collection fusion problem", Proceeding of the 3rd Text Retrieval Conference (TREC-3). NIST Special Publication 500-225, 1994.

[3] Ellen M. Voorhees, Narenda K.Gupta, and Ben Johnson-Laird. "Learning collection fusion strategies", proceeding of 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 172-179, 1995.

[4] Susan T. Dumais "LSI meets TREC: A status report", Proceeding of the First Text Retrieval Conference (TREC-1), pages 137-152, NIST Special Publication 500-207, 1993.

[5] Ellen M.Voorhees "Siemens TREC-4 Report: Futher Experiments with Database Merging", Proceeding the 4th Text Retrieval Conference (TREC-4). NIST Special Publication 500-236, 1995.

[6] R.R. Yager. "On the Fusion of Documents from Multiple Collection information Retrieval Systems", Journal of American Society for Information Science, 1998.

[7] D. Harman "Overview of the Third Text Retrieval Conference (TREC-3)" NIST Special Publication 500-226 , 1995.

[8] Chris Buckley. "Implementation of the SMART information retrieval system" Technical Report 85-686, Computer Science Department, Cornell University, 1985.

[9] J.H. Lee, J.S. Ahn, "Using n-Grams for Korean Text Retrieval", Proceeding of the 19th Annual International ACM SIGIR Conference on Research and Delevopment in Information Retrieval, 1996.