FDC-TCT를 이용한 웹 문서 클러스터링 성능 개선 기법

고 석 범[†] · 윤 성 대[††]

요 약

키워드를 통한 웹 검색 결과의 분류와 같은 후처리가 요구되는 문서 분류 문제에서, 기존의 문서 분류 또는 클러스터링 알고리즘을 적용하는 데에는 많은 문제가 있다. 그 중에서 고려해야 할 가장 심각한 두 가지 문제가 있다. 첫째는 전문가가 관여하여 범주를 선정하는 문제이고, 둘째는 문서분류에 소요되는 수행시간이 긴 문제이다. 따라서 본 논문에서는 이행적 폐쇄 트리를 이용하여 문서 유사도 계산 횟수를 크게 줄이고, 정확도의 희생을 최소화하면서 신속한 처리가 가능한 새로운 웹 문서 클러스터링 기법을 제안하다. 또한, 제안된 기법의 효율성을 검증하기 위하여 기존의 알고리즘과 비교 평가 및 분석한다.

키워드 : 웹 문서 클러스터링, 텍스트마이닝, 이행적 폐쇄

A performance improvement methodology of web document clustering using FDC-TCT

Sucbum Ko[†] · SungDae Youn[††]

ABSTRACT

There are various problems while applying classification or clustering algorithm in that document classification which requires post processing or classification after getting as a web search result due to any keyword. Among those, two problems are severe. The first problem is the need to categorize the document with the help of the expert. And, the second problem is the long processing time the document classification takes. Therefore we propose a new method of web document clustering which can dramatically decrease the number of times to calculate a document similarity using the Transitive Closure Tree(TCT) and which is able to speed up the processing without loosing the precision. We also compare the effectivity of the proposed method with those existing algorithms and present the experimental results.

Key Words : Web Document Clustering, Text Mining, Transitive Closure

1. 서 론

웹의 비약적인 성장과 더불어, 웹문서를 통한 정보의 획득 비율이 높아지고 있다. 그러나 대량의 웹문서로부터 원하는 정보를 획득하는 것은 난해한 문제이다. 기존의 포털 사이트에서 키워드 검색을 통해 획득되는 검색 결과는 대량의 정보로서 검색자가 필요한 정보를 다시 분류해야 하는 문제가 있다. 이러한 이유에서 신속하고 정확하게 관련된 문서끼리 분류하는 기법에 대한 연구가 필수적이다[2, 5].

문서를 분류하는 방법에는 크게 전문가에 의해 미리 정해진 범주(Category)에 따라 문서를 포함시키는 문서 범주화(Document categorization) 기법과, 어떠한 사전적인 범주를 선정하지 않고 단지 문서들 간의 유사도를 통해 자체적으로 분류하는 문서 클러스터링(Document clustering) 기법이 있다[10, 12]. 문서 범주화 기법은 문서 클러스터링 방법에 비해 대개 정확도(Precision)는 높으나, 범주의 선정과 같은 전문가의 개입이 필요하므로 실제 적용이 어려운 문제가 있다. 반면에 문서 클러스터링 방법은 전문가의 개입과 같은 전제조건을 요구 하지 않지만, 대체로 문서 범주화 기법보다 수행 속도가 매우 느린 문제가 있다[12]. 그리고 문서 클러스터링 기법에서도, 정확도 및 응답도(Recall)는 낮으나 수행속도가 빠른 반복적 클러스터링(Iterative clustering) 기법이 있고, 역으로 정확도 및 응답도는 높으나 수행 속도가 현저하게 느린 계층적 집괴 클러스터링(Hierarchical agglomerative clustering)기법이 있다.

따라서 본 논문에서는 정확도의 희생을 최소화하면서 신속한 문서 클러스터링이 가능한 기법을 제안한다. 즉, 제안하는 문서 클러스터링 기법은 전문가의 범주 분류와 같은 사전 요구 사항이 필요하지 않고, 정확도 및 응답도의 성능

※ 이 논문은 2004년도 두뇌한국21(BK21) 사업에 의해 지원되었음.
† 준 회 원 : 부경대학교 대학원 전자계산학과
†† 정 회 원 : 부경대학교 전자계산학과 교수
논문접수 : 2005년 2월 28일, 심사완료 : 2005년 4월 4일

이 우수한 계층적 집괴 클러스터링 기법의 구조를 가지면서, 수행속도를 향상시키기 위한 방법이다.

본 논문의 구성은, 2장에서 문서 범주화 및 클러스터링 기법에 대한 기존 연구들에 관하여 기술하며, 본 연구의 필요성에 대하여 언급한다. 3장에서는 본 연구에서 제안하는 새로운 문서 분류기법인 FDC-TCT에 관하여 자세히 기술하고, 4장에서는 실험을 통해 기존의 문서 클러스터링 기법들과 비교 분석한다. 끝으로 5장에서 본 연구에 대한 결론과 향후과제에 관하여 논한다.

2. 관련 연구

방대한 문서로부터 문서의 유사성에 따라 문서를 분류하는 방식은 크게 두 가지가 있다. 첫째는 문서 범주화 기법이다. 이 기법은 사전에 전문가에 의해 범주를 구분하여 두었다가, 분류 대상 문서의 요청에 대해 분류기가 가장 적합한 범주에 문서를 포함하는 방식이다. 문서 범주화 기법은 정확성은 높으나, 범주 선정을 위해 지속적인 전문가의 개입이 필요한 문제가 있다[11, 16]. 그러므로 이 기법은 다양하고 새로운 범주가 자주 발생하는 웹 검색결과 분류와 같이 후처리가 필요한 분류에는 적절하지 않은 기법이다. 두 번째 방식은 문서 클러스터링 기법이다. 문서 클러스터링 기법은 자동문서 분류 기법처럼 사전에 전문가의 개입에 의한 범주 분류가 필요 없다. 각 문서들을 독립적인 개체로 인식하여 각 문서간의 유사도를 측정하여 유사한 문서끼리 그룹화하여 클러스터를 생성하는 것이다. 이 기법의 단점은 낮은 정확도와 방대한 수행시간이다.

대표적인 자동문서 분류 기법에는 K-NN 기법[3, 11]과 베이지안 분류(Bayesian classification)[16]가 있다. 그리고 문서 클러스터링 기법은 다시 반복적 클러스터링 기법과 계층적 집괴 클러스터링 기법으로 나눌 수 있다[12, 14].

반복적 클러스터링 기법은 비계층적 클러스터링 기법으로서 특정 개수의 클러스터링을 목표로 반복적으로 클러스터의 중심(Centroid)을 갱신하면서 관련 있는 문서를 클러스터링하는 기법이다. 반복적 클러스터링 기법의 대표적인 예는 k-mean 알고리즘이다[4, 7, 13].

계층적 집괴 클러스터링은 하나의 문서를 클러스터로 간주하여 각 문서간의 유사도를 바탕으로 클러스터링하는 방법으로서, 단일 링크(Single Link), 완전 링크(Complete Link), 집단평균 링크(Group Average Link)이 있다.

대개 계층적 집괴 클러스터링은 반복적 클러스터링에 비해 수행속도는 느리지만, 정확도가 높은 특성이 있다[12, 14]. 따라서 본 논문에서는 계층적 집적 클러스터링 기법과 같이 정확도를 최대한 유지하면서 수행속도가 느린 문제점을 해결 할 수 있는 클러스터링 기법을 제안하는 것이다.

2.1 문서 클러스터링 기법

문서 클러스터링 기법은 자동문서 분류기법과는 달리 전문가의 개입을 요구하지 않고 자체적으로 유사한 문서끼리 분류함으로써, 실세계의 문서 분류시스템의 구현에 적절한 기법이다. 문서 클러스터링 기법에는 크게 반복적 클러스터링과 계층적 집괴 클러스터링 기법이 있다.

2.1.1 반복적 클러스터링(Iterative clustering)

반복적 클러스터링 기법의 대표적인 기법은 k-mean 알고리즘이다. k-mean 알고리즘의 주요 생성 과정은 k개의 클러스터를 목표로, 각 centroid를 중심으로 각 문서와의 거리를 구하여 클러스터링하고, 생성된 클러스터의 centroid를 각각 갱신하여 반복적으로 클러스터링하여, centroid의 변화가 임계치 이하일 때 종료하는 기법이다[13]. k-mean 알고리즘은 구현이 간단하고 계산 복잡도가 작은 알고리즘이지만 계층적 집괴 클러스터링에 비해 정확도 및 응답도(Recall)가 떨어지는 특성이 있다.

2.1.2 계층적 집괴 클러스터링(Hierarchical agglomerative clustering)

계층적 집괴 클러스터링은 클러스터링 하기 위한 각 문서들을 하나의 독립된 클러스터로 간주하여, 각 클러스터간의 유사도(Similarity)를 계산하여 점차적으로 클러스터를 병합하는 기법이다. 계층적 집괴 클러스터링은 클러스터의 유사도를 결정하는 방법에 따라 단일링크, 완전링크, 집단평균링크로 구분한다[12].

(1) 단일 링크(Single Link : SL)

단일링크의 클러스터 유사도는 클러스터에 포함되어 있는 문서들간의 유사도 중에서 최대 유사도를 클러스터의 유사도로 결정하는 방법이다.

$$SL(C_i, C_j) = \max_{d_i \in C_i, d_j \in C_j} \{\cos(d_i, d_j)\}$$

where, C_i: i 번째 클러스터
d_i: i 번째 문서
$\cos(d_i, d_j)$: 문서 d_i와 d_j간의 코사인 유사도

(2) 완전 링크(Complete Link : CL)

완전링크의 클러스터 유사도는 클러스터에 포함되어 있는 문서들간의 유사도 중에서 최소 유사도를 클러스터의 유사도로 결정하는 방법이다.

$$CL(C_i, C_j) = \min_{d_i \in C_i, d_j \in C_j} \{\cos(d_i, d_j)\}$$

(3) 집단 평균 링크(group Average Link : AL)

집단 평균 링크에서 클러스터의 유사도는 클러스터에 포함되어 있는 문서들 간의 유사도의 평균을 클러스터의 유사도로 결정하는 방법이다.

$$AL(C_i, C_j) = \frac{1}{n_i n_j} \cdot \sum_{d_i \in C_i, d_j \in C_j} \cos(d_i, d_j)$$

where, n_i: 클러스터 i에 포함된 문서들의 총수

계층적 집괴 클러스터링은 미리 정한 클러스터의 개수까지 병합되거나, 임의의 클러스터 응집도 이상이 될 때까지 수행한다. 계층적 집괴 클러스터링은 클러스터에 속하는 모든 문서간의 유사도를 계산해야 하므로, 문서분류의 정확도는 우수하나 과도한 수행시간이 걸리는 단점이 있다.

2.2 이행적 폐쇄(Transitive Closure)

데이터베이스의 두 레코드간의 비교에서 이행적 폐쇄가 [1]과 [6]에서 적용이 되었다. 이행적 폐쇄 이론의 핵심은 레코드 a와 레코드 b가 유사하고, 레코드 b와 레코드 c가 유사하다면, 레코드 a와 c간의 유사도를 계산하지 않고도 레코드 a와 c는 유사하다고 간주하는 것이다[9].

이러한 이행적 폐쇄를 문서 클러스터링 기법에 적용하면, 문서 집합에서 유사도 계산 회수를 대폭적으로 줄일 수 있지만, 정확도가 손실되는 단점이 있다. 정확도의 희생을 최소화하기 위해서는 문서 비교를 세밀하게 조절 할 수 있는 기법이 필요하다.

3. FDC-TCT(Fast Document Clustering Based Transitive Closure Tree) 설계

본 논문에서 제안하는 문서분류 시스템은 자동화된 문서 클러스터링 기법으로서, 범주 선정과 같은 전문가의 개입을 필요로 하지 않으며, 반복적 클러스터링보다 정확도를 향상시키고 계층적 집괴 클러스터링 기법보다 수행 시간의 단축을 목적으로 설계한다.

본 논문에서 제안하는 이행적 폐쇄트리 기반 문서 클러스터링(Fast Document Clustering based Transitive Closure Tree : FDC-TCT) 기법은 원 문서로부터 명사 추출 및 불용어 처리 과정을 통해 키워드를 추출한 문서들을 이진트리로 구성하여 클러스터링한다. 문서 클러스터링하는 주요 과정은 연관순서에 따라 문서들을 재정렬하고, 문서의 순서에 따라 이진트리로 구성하여 링크에 연결된 문서들 끼리 유사도를 계산한다. 사전에 정한 임의의 임계치보다 큰 문서를 유사하다고 가정하여 유사 링크를 연결한다. 이후의 문서간의 비교는 Sibling Comparison(SC), Descendant Comparison(DC), Ancestor Comparison(AC)을 수행하여 문서 클러스터를 형성한다.

3.1 문서 클러스터링 요소

본 절에서는 문서 클러스터링에 필요한 요소들에 대하여 기술한다. 우선 FDC-TCT에 대해 기술하기 위해 사용되는 파라메터들은 다음과 같다.

- D : 문서집합으로 $D=\{d_1, d_2, \cdots, d_n\}$
- d_i : 문서집합 D에서의 i번째 문서
- t_j : 임의의 문서에서의 j번째 단어
- C_i : i 번째 클러스터로서, $C_i \subset D$
- $minS$: 최소 문서 유사도로서, $0 \le minS \le 1$
- $minC$: 최소 응집도로서, $0 \le minsC \le 1$
- $minMer$: 최소 병합도로서, $0 \le minMer \le 1$
- $|d_i|$: 문서 i에 포함된 유일한 단어의 총 개수
- $|d_i \cap d_j|$: 문서 i와 j에 공통으로 포함된 유일한 단어의 총 개수
- $|C_i|$: 클러스터 i에 포함된 문서의 총 개수

본 연구에서 문서의 특징을 나타내는 키워드 산출을 위한 가중치를 구하기 위하여, 보편적으로 널리 사용되는 TFIDF (Term Frequency Inversed Document Frequency) 함수를 적용하였다[8]. 즉, 문서 d_i에서 단어 t_j의 가중치 w_{ij}는 식 3.1과 같다.

$$w_{ij} = TF_{ij} \cdot \ln \frac{n}{IDF_j} \tag{3.1}$$

여기에서, TF_{ij}는 문서 d_i에서 단어 t_j의 출현 빈도수이고, IDF_j는 전체 문서 수 n에 대하여 단어 t_j의 출현빈도수이다. 그러므로 임의의 키워드가 문서에 포함되기 위해서는 임의의 키워드가 문서내에서는 빈번하게 출현하고, 다른 문서에 대해서는 가능한 적게 출현해야 한다.

3.1.1 문서 유사도(Document similarity)

문서 유사도를 측정하기 위한 수식은 식 3.2와 같다.

$$S(d_i, d_j) = \frac{|d_i \cap d_j|}{\min(|d_i|, |d_j|)} \tag{3.2}$$

식 3.2는 계산 시간이 많이 소요되는 기존의 벡터 기반의 유사도 계산보다 간단하여 빠른 계산이 가능한 방법이다. 분모는 문서 d_i와 d_j간의 유사도 정도를 나타내며, 분자는 문서에 포함된 키워드의 개수가 작은 문서를 기준으로 유사도를 측정함으로써, 긴 문서가 짧은 문서와의 유사도 비교에 정당성을 반영하는 정규화 효과를 부여하기 위함이다.

3.1.2 클러스터 응집도

클러스터 응집도는 임의의 클러스터 내부의 구성 문서들 간의 관련성의 정도를 나타내며 식 3.3와 같다.

$$C(C_i) = \frac{\sum_{d_i \in C_i,\, d_j \in C_i - \{d_i\}} S(d_i, d_j)}{|C_i|} \tag{3.3}$$

식 3.3과 같이 클러스터 C_i의 클러스터 응집도는 클러스터 내부의 문서들의 유사도 평균으로 구한다. 클러스터 응집도는 문서나 클러스터를 병합할 때 클러스터가 일정 응집도 이상을 항상 유지하는 역할을 한다.

3.2 이행적 폐쇄 트리(Transitive Closure Tree)

이행적 폐쇄는 대량의 문서의 유사도 계산 횟수를 대폭적으로 줄일 수 있는 방법이다. 그러나 문서 클러스터링에서 이행적 폐쇄의 적용은 정확도를 감소시킨다. 이러한 정확도의 희생을 최소화하기 위해서, 클러스터가 항상 일정한 유사도를 유지시키기 위한 조정이 필요하다.

본 논문에서는 문서 클러스터링을 위해 이행적 폐쇄 추론을 이진트리에 적용한 이행적 폐쇄 트리를 제안한다. 이행적 폐쇄 트리 TCT는 다음과 같이 구성된다.

$$TCT = \{ V, E \}$$
$$V = \{ v_1, v_2, \cdots, v_n \}$$
$$E = \{ e_{12}, e_{13}, e_{24}, e_{25}, \cdots, e_{i(2i)}, e_{i(2i+1)} \}$$

각 노드 v_i는 정렬된 문서 집합 D로부터 문서 d_i를 의미한다. 또한, 두 노드간의 링크 e_{ij}는 문서 d_i와 d_j간의 문서 유사도 $S(d_i, d_j)$를 의미한다. TCT의 자료구조는 (그림 1)과 같이 구성한다. (그림 1)과 같이 TCT는 다중링크 트리 형태로 구성되어, LC와 RC는 실선으로 각각 왼쪽 자식과 오른쪽 자식을 가리키고, SLC와 SRC는 점선으로 자신과 유사한 자식을 가리킨다. 즉, SLC와 SRC의 점선 링크는 두 문서간의 유사도가 $minS$보다 크다는 것을 의미한다.

3.3 문서 비교 패턴 알고리즘

문서 집합으로부터 문서를 비교하는 패턴은 (문서 vs. 문서), (문서 vs. 클러스터), (클러스터 vs. 클러스터) 세 가지 경우로 구분할 수 있다.

3.3.1 문서 vs. 문서

문서와 문서간의 비교로서 식 3.3에서 정의한 문서 유사도를 통해 문서의 병합 여부를 결정할 수 있다. 즉, 미리 정한 임의의 최소 유사도 이상의 경우에 두 문서는 병합되어 클러스터를 형성한다.

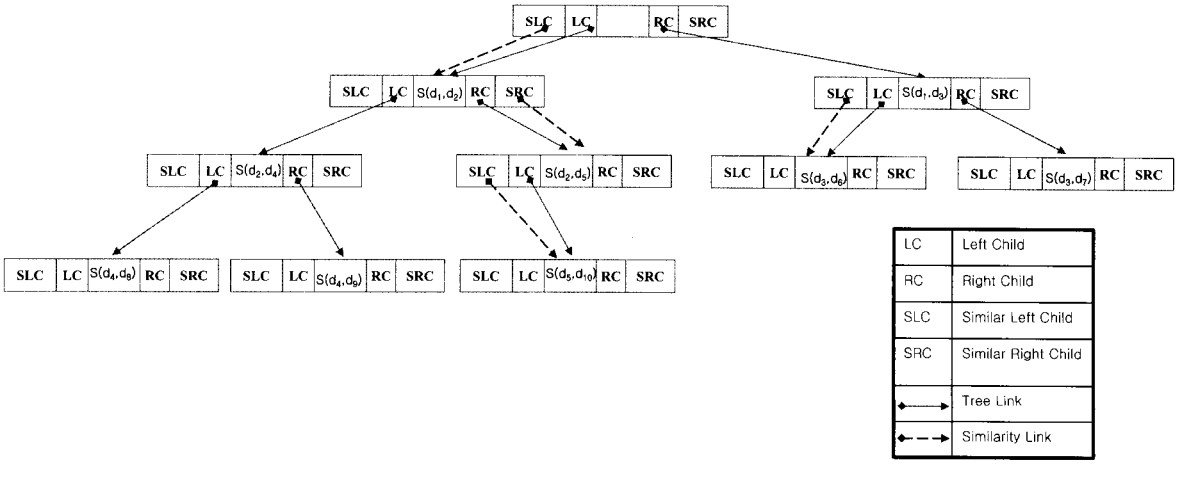
Procedure DocToDoc
begin
calculate $S(d_i, d_j)$;
if ($S(d_i, d_j) \geq$ minS)
then mergeDTD(d_i, d_j);
end

3.3.2 문서 vs. 클러스터

임의의 문서와 두 개 이상의 유사 문서로 결합된 클러스터와의 비교로서, 본 논문에서는 집단 평균 연결을 사용하여 문서의 병합 여부를 결정한다. 문서와 클러스터간의 병합과정은 다음의 과정에 따라 수행된다.

Procedure DocToCluster
set the document d_d as the compared document;
set the document d_c as the compared document with d_d in the cluster C_a;
begin
calculate $C(C_a) = \frac{\sum_{k \neq l \text{ and } k,\, l \in C_a} S(d_k, d_l)}{|C_a|}$;
calculate the document similarity between d_d and d_c:
$SA_{dc} = \frac{S(d_d, d_c) + C(C_a)}{2}$;
calculate the cluster coherence:
$C(C_a \cup d_d) = \frac{\sum_{d_i, d_j \in C_a} S(d_i, d_j)}{|C_a| + 1}$;
if ($SA_{dc} \geq$ minS and $C(C_a \cup d_d) \geq$ minC)
then mergeDTC(d_d, C_a);
set a next comparison node as a last node of the cluster;
end

문서와 클러스터의 비교에서의 주요 고려사항은 문서가 클러스터에 병합될 때 단순히 클러스터의 특정 문서와의 문서 유사도를 계산하여 병합 여부를 결정하는 것이 아니라, 클러스터에 속하는 전체 문서 유사도의 평균값을 고려하여



(그림 1) FDC-TCT 자료구조

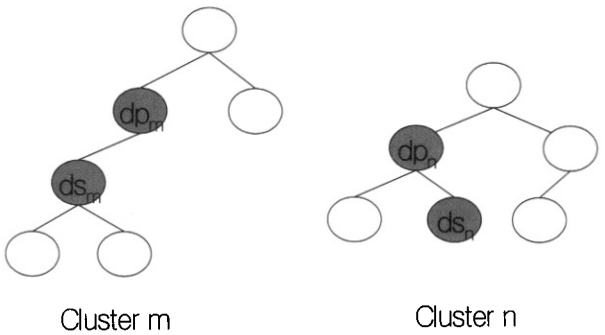
통합하는 것이다. 즉, 비교되는 문서의 유사도는 클러스터에 포함되는 문서들과의 전체 평균이 된다. 또한, 병합 여부의 결정에서는 계산된 유사도가 최소 유사도 이상이고 병합 후에 생성되는 클러스터의 응집도가 최소 응집도 이상이 되어야 문서가 클러스터에 병합된다.

3.3.3 클러스터 vs. 클러스터

두개 이상의 유사 문서로 결합된 클러스터간의 비교로서, 다음의 과정을 통해 병합을 수행한다.

Procedure ClusterToCluster
set two clusters to be compared as C_n and C_m, respectively;
begin
calculate $C(C_n)$, $C(C_m)$;
find a $dp_m - ds_m$ link which approximates $C(C_m)$ in C_m;
find a $dp_n - ds_n$ link which approximates $C(C_n)$ in C_n;
construct D_F={ dp_m, ds_m, dp_n, ds_n};
calculate $SC = \frac{\sum_{d_i \in D_F,\ d_j \in D_F - \{d_j\}} S(d_i, d_j)}{6}$;
if ($SC \geq$ minMer and $C(C_n \cup C_m) \geq$ minC)
then mergeCTC(C_n, C_m);
set the next comparison node to the last node of cluster;
end

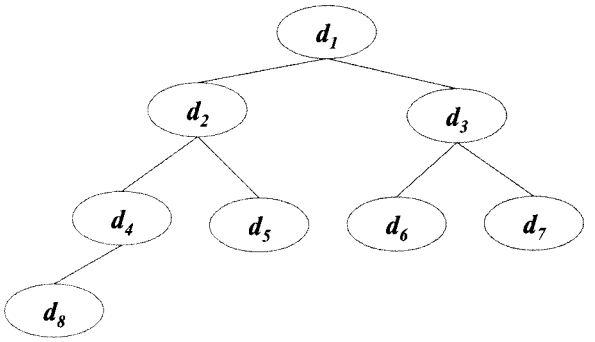
클러스터와 클러스터의 병합에서는, (그림 2)와 같이 두 클러스터의 평균 유사도와 가장 근사한 각각의 링크 $dp_m - ds_m$, $dp_n - ds_n$을 찾아 D_F를 구성한다. 그러고 나서 두 클러스터의 연결 유사도 SC를 D_F에 포함되는 6개의 문서간의 유사도의 평균으로 하여 최소병합도 *minMer* 보다 클 때 두 클러스터는 병합된다. 그 이유는 적은 개수의 문서로 구성된 클러스터가 다수의 문서로 구성된 클러스터에 강제적으로 병합되는 것을 막기 위함이다.



(그림 2) 클러스터 m과 클러스터 n의 병합

3.4 유사도 비교 알고리즘

FDC-TCT는 Initial Comparison, Sibling Comparison, Descendant Comparison, Ancestor Comparison을 수행하면서 유사한 문서끼리 클러스터링한다.



(그림 3) Initial Comparison

3.4.1 Initial Comparison

(그림 3)과 같이 TCT에서 각 문서들은 부모-자식 페어들간에 유사도를 계산하여 *minS*에 따라 병합여부를 결정한다.

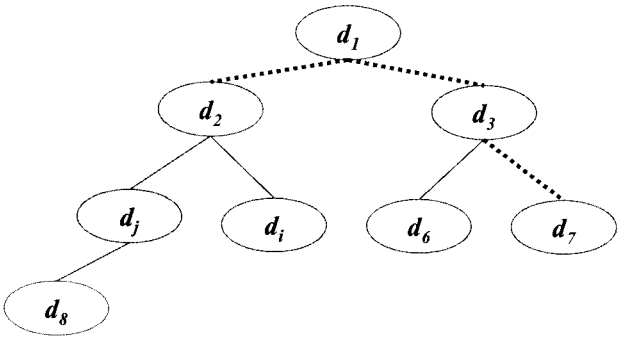
Step 1 : execute BFS in TCT
Step 2 : set d_i and d_j as the current visiting node and its parent node, respectively
Step 3 : if the comparison pattern of d_i and d_j is the document vs the document, then call procedure DocToDoc
Step 4 : if the comparison pattern of d_i and d_j is the document vs the cluster, then call procedure DocToCluster
Step 5 : if all nodes in TCT are visited, then terminate the procedure
Step 6 : visit the next node and go to Step 2

위의 프로시저와 같이, TCT에서 BFS을 통해 방문하는 노드 d_i와 부모노드 d_j의 유사도를 계산하기 위하여 문서 비교 패턴에 따라 합당한 프로시저를 호출하여 문서의 병합 여부를 결정한다. 즉, 비교하는 문서 비교 패턴이 (문서 vs. 문서)이면 3.3절에서 정의한 DocToDoc 프로시저를 호출한다. 그리고 나서 모든 노드를 방문하면 프로시저를 종료하고 그렇지 않으면 BFS를 통해 다음 노드를 방문하여 Step 2를 수행한다.

3.4.2 Sibling Comparison

트리 레벨별로 유사하지 않는 자식 노드간의 유사도 비교이다.

Step 1 : execute BFS in TCT
Step 2 : set d_i and d_j as the current visiting node and its sibling node, respectively
Step 3 : if d_i and its parent node is not similar and d_j and its parent node is not similar, then call a proper procedure to merge d_i and d_j accord-



(그림 4) Sibling Comparison

ing to the document comparison pattern

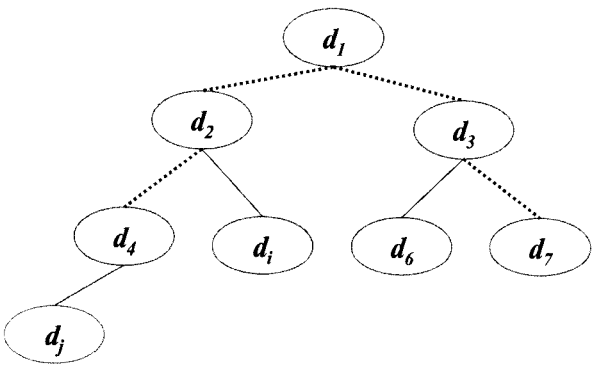
Step 4 : if all nodes in TCT are visited, then terminate the procedure

Step 5 : visit the next node and go to Step 2

Sibling Comparison을 수행하면 각 레벨을 기준으로 클러스터들이 생성되거나, 어떤 클러스터에도 소속되지 않는 기아노드가 남게 된다.

3.4.3 Descendant Comparison

TCT에서 임의의 노드에서 후손(하위) 방향으로 유사하지 않는 노드나 클러스터를 찾아 유사도를 검사하기 위한 비교이다.



(그림 5) Descendant Comparison

Step 1 : execute BFS in TCT

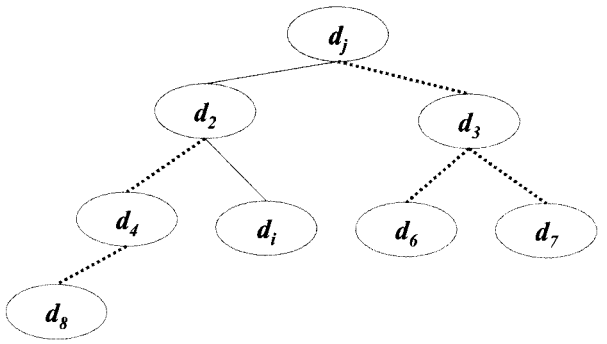
Step 2 : set d_i as the current visiting node

Step 3 : if d_i and its parent node is not similar, then find d_j, the first node that is not similar its parent node in descendant nodes, otherwise go to step 5

Step 4 : call a proper procedure to merge d_i and d_j according to the document comparison pattern

Step 5 : if all nodes in TCT are visited, then terminate procedure

Step 6 : visit the next node and go to Step 2



(그림 6) Ancestor Comparison

3.4.4 Ancestor Comparison

TCT에서 임의의 노드에서 조상(상위) 방향으로 유사하지 않는 노드나 클러스터를 찾아 유사도를 검사하기 위한 비교이다.

Step 1 : execute BFS in TCT

Step 2 : set d_i as the current visiting node

Step 3 : if d_i and its parent node is not similar, then find d_j, the first node that is not similar its ancestor node in ancestor nodes, otherwise go to step 5

Step 4 : call a proper procedure to merge d_i and d_j according to the document comparison pattern

Step 5 : if all nodes in TCT are visited, then terminate the procedure,

Step 6 : visit the next node and go to Step 2

위에서 기술한 비교 패턴에 따라 FDC-TCT의 전체적인 수행 알고리즘은 다음과 같다.

```
Procedure TCT_Clustering
begin
 execute the initial comparison;
 execute the sibling comparison;
 set L-TCT as a left sub-tree of root node;
 set R-TCT as a right sub-tree of root node;
 FORALL nodes in TCT DO
  execute the descendant comparison for L-TCT;
  execute the descendant comparison for R-TCT;
 ENDFOR
 FORALL nodes in TCT DO
   execute the ancestor comparison from the last terminal node
to a root node;
 ENDFOR
end
```

4. 성능 평가

본 연구에서 제안한 FDC-TCT 알고리즘의 우수성을 입

〈표 1〉 실험 환경

프로그램	명사 추출	HAM[15] 사용
	텍스트 변환	자작 프로그램 (VC++ 6.0)
	문서 클러스터링	자작 프로그램 (JDK 1.3)
수행 환경	PC	P-III 800MHz, 512M RAM
문서 데이터	네이버 디렉토리검색 (검색어: '멀티미디어')	607개 문서, 62개 카테고리
parameter	*minS*=0.7, *minC*=0.6, *minMer*=0.4, k=62	

증하기 위하여 성능평가를 수행한다. 비교 대상 알고리즘은 반복적 클러스터링 기법으로는 k-mean 알고리즘을, 계층적 집괴 클러스터링 기법으로는 SL, CL, AL 기법을 선정한다. 각 기법에 대하여 정확도, 응답도, F-Measure, 수행시간을 측정하여 분석한다.

우선 실험에 사용되는 문서는 네이버 디렉토리 검색에서 '멀티미디어' 카테고리에 소속되는 문서집합으로서 각 하위 카테고리를 하나의 클러스터로 간주하여 실험하였다. 실험 환경을 <표 1>에 요약하였다.

정확도와 응답도를 계산하기 위해 디렉토리 검색에서 제공하는 문서들에 대하여 다음과 같이 문서 레벨링을 구성하였다.

the lowest label number	category_id	document_id

여기에서, 'the lowest label number'는 임의의 문서가 최상위 레벨이 1인 카테고리의 레벨에서 최하위 카테고리를 의미한다. 그리고 임의의 문서가 해당 레벨에서 몇 번째 카테고리에 속하는지 category_id로 표기하고, 해당 카테고리에서 몇 번째 문서인지 document_id로 표기한다. 예를 들어 (그림 7)과 같은 네이버에서 '멀티미디어' 디렉토리 검색어에 대한 카테고리 집합에서, '그래픽' 카테고리에 속하는 문서들은 '2 1 document_id'가 되고, '그래픽 디자이너' 카테고리에 속하는 문서들은 '3 2 document_id'가 된다. 문서 레벨링을 통해 두 문서가 동일 카테고리에 속하는지 여부는 간단한 마스크 연산으로 판별 할 수 있다.

따라서, 정확도와 응답도를 구하기 위하여 다음과 같이 FDC-TCT가 분류한 대표 클러스터를 구한다.

Step 1 : 분류기가 분류한 클러스터별로 문서 d_i가 어느 카테고리에 속하는지 계산

$$LC(d_i) = LCD(d_i)\ AND\ [1111\ 1111\ 11111111\ 0000000000000000],$$
$$\text{for } i = 1, 2, \cdots, n$$
$$\textit{where } LCD(d_i): \text{문서 } d_i \text{의 레벨링}$$
$$n: \text{문서의 총수}$$

Step 2 : 분류기가 분류한 클러스터별로 소속 문서가 제일 많은 대표 클러스터를 구한다.

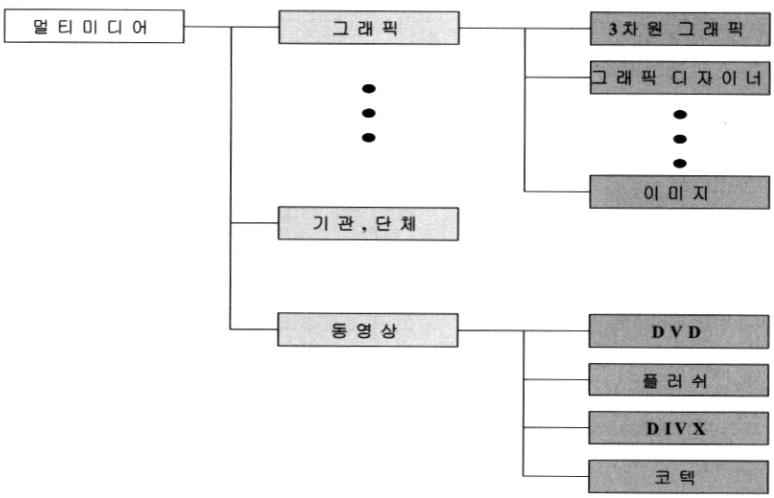
$$RC_j = \max\{|LC(d_i)|\,|\,i \neq k,\ LC(d_i) = LC(d_k)\},$$
$$\text{for } i, k = 1, 2, \cdots, n,\ j = 1, 2, \cdots, m$$
$$\textit{where, } m: \text{분류된 클러스터의 총수}$$

그리고 나서, 다음과 같이 각 클러스터별로 정확도 p_j를 각각 계산하고,

$$p_j = \frac{|RC_j|}{|OC_j|},\ j = 1, 2, \cdots, m$$
$$\textit{where, } |OC_j|: j \text{ 카테고리에 포함된 문서의 총수}$$

다음 수식과 같이 p_j의 평균을 정확도 P로 한다.

$$P = \frac{\sum_{j=1}^{m} p_j}{m},\ j = 1, 2, \cdots, m$$



(그림 7) '멀티미디어' 키워드에 대한 네이버 디렉토리에서의 카테고리 트리

그리고 응답도를 구하기 위하여 각 클러스터별 응답도 r_j를 다음 수식을 통하여 구하고

$$r_j = \frac{|RC_j|}{|CD_j|},\ j=1, 2, \cdots, m$$

where, $|CD_j|$: 분류기가 분류한 j번째 클러스터에 포함된 문서의 총수

다음 수식을 통해 r_j 평균을 구하여 응답도 R을 구한다.

$$R = \frac{\sum_{j=1}^{m} r_j}{m},\ j=1, 2, \cdots, m$$

끝으로, 다음 수식에 의해 F-Measure를 측정 할 수 있다.

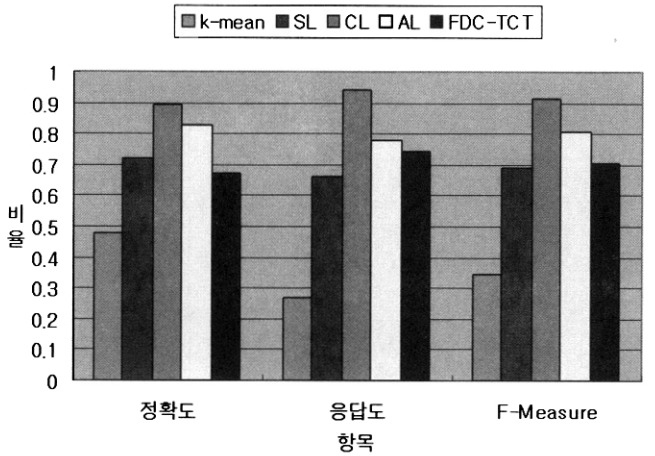
$$F = \frac{2 \times P \times R}{P+R}$$

본 논문에서 제안하는 FDC-TCT 알고리즘과 비교하는 기존 알고리즘으로는 문서 클러스터링 기법중, 반복적 클러스터링에는 k-mean, 계층적 집괴 클러스터링에서는 단순링크(SL), 완전링크(CL), 집단평균링크(AL)기법을 선정하였다. 607개의 문서에 대해 정확도, 응답도, F-Measure를 측정한 결과는 (그림 8)과 같다.

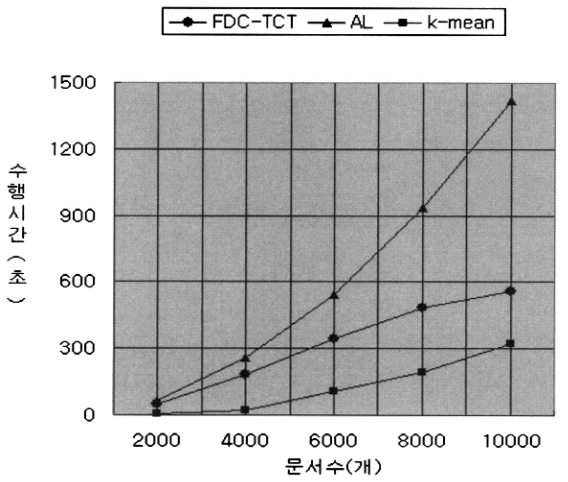
(그림 8)과 같이 FDC-TCT는 계층적 집괴 클러스터링 기법 보다는 정확도 및 응답도가 다소 저조한 반면 k-mean 알고리즘 보다는 우수한 특성이 있다. 또한 FDC-TCT는 정확도 보다 응답도가 탁월한 장점이 있는데 이는 이행적 폐쇄 이론에 따라 유사도 계산 측면에서 엄격하지 않은 특성 때문이다.

수행시간 비교에서는 문서의 개수를 2000, 4000, 6000, 8000, 10000개의 문서들을 구성하여 k-mean, 집단평균링크(AL), FDC-TCT와 비교 하였다. 계층적 집괴 알고리즘 중 F-Measure 면에서 비교적 중간정도의 성능을 갖는 AL기법을 선정하였다.

(그림 9)와 같이 수행시간면에서는 문서간 비교회수가 가장 적은 k-mean 알고리즘이 가장 월등한 성능을 보이는 반



(그림 8) 정확도, 응답도, F-Measure의 비교



(그림 9) 수행 시간 비교

면, 집단평균링크(AL) 기법이 가장 긴 수행시간을 필요로 한다. FDC-TCT 기법은 k-mean 알고리즘에 비해 다소 긴 수행시간이 필요하지만, AL 기법에 비해서는 문서의 수가 많을수록 더욱 빠른 성능을 갖는다. 이것은 이행적 폐쇄 이론에 따라 문서의 비교 회수가 Initial Comparison 단계에서 $nlogn$번이 필요하고, 이후의 단계에서는 클러스터의 개수에 따라 급격하게 비교회수가 줄어드는데 반하여, AL 기법에서는 최소한 $n^2\log n$번의 비교회수가 필요하기 때문이다.

이상과 같이 실험을 통해 본 논문에서 제안한 FDC-TCT 기법은 정확도 및 응답도가 저조한 k-mean 알고리즘의 단점과, 수행속도가 저조한 계층적 집괴 클러스터링 기법의 단점을 극복하였음을 입증하였다.

5. 결 론

본 논문에서는 웹 검색 결과 분류와 같이 일정 수준의 정확도 및 응답도를 요구하면서 신속한 문서 클러스터링이 가능한 FDC-TCT 기법을 제안하였다. FDC-TCT는 전문가의 개입을 요구하지 않는 문서 클러스터링 방법으로서, 기존의 정확도 및 응답도가 우수한 계층적 집괴 클러스터링 기법의 장점을 보유하면서도 저속의 수행속도의 단점을 개선하는 방법이다.

그래서 본 논문에서는 이행적 폐쇄 이론을 접목한 이행적 폐쇄트리를 구성하여 문서간 유사도 계산 회수를 대폭적으로 감소시켰다. 그리고 이에 상응하는 정확도 및 응답도의 손실을 줄이기 위하여 문서 비교패턴 및 유사도 비교 알고리즘을 제안하였다.

제안한 FDC-TCT의 효율성을 검증하기 위한 실험을 통하여 F-Measure는 AL 기법에 비하여 *12%* 감소하였으나, k-mean 알고리즘에 대해서는 *203%* 향상되었고, 수행속도면에서는 AL 기법보다 198% 향상 된 것을 알 수 있다. 특히, FDC-TCT의 수행시간의 성능은 문서수의 증가에 따라 더 좋은 결과를 갖는다.

그리고 본 논문에서 제안한 FDC-TCT 기법은 웹 검색 결과 분류뿐만 아니라 웹문서, e-mail, 신문기사 분류와 같이 텍스트기반 문서 분류기법에 활용 할 수 있다.

향후연구 과제로는 단어 기반으로 유사도를 계산하여 클러스터링하는 것이 아니라 구(phrase) 단위로 빠른 문서 클러스터링을 수행하는 기법에 대한 연구가 필요하다. 그리고 클러스터링 후 생성된 TCT 정보를 이용한 점증적인 클러스터링 기법에 관한 연구가 필요하다.

참 고 문 헌

[1] A. E. Monge and C. P. Elkan, "An efficient domain-independent algorithm for detecting approximately duplicate database records", Proceeding of the ACM SIGMOD workshop on research Issues on knowledge discovery and data mining, pp.125-130, 1997.

[2] Hwanjo Yu, Jiawei Han, and Kevin C. Chang, "PEBL: Web Page Classification without Negative Examples", IEEE tran. on knowledge and data engineering, Vol.16, No.1, pp.70-81, Jan., 2004.

[3] J. C. Song and J. Y. Shen, "A web document clustering algorithm based on concept of neighbor", Proceedings of the second international conference on machine learning and cybernetics, Xi'an, 2-5, pp.46-50, Nov., 2003.

[4] Khaled Alsabti, et al, "An efficient K-Means Clustering Algorithm", IIPS 11th International Parallel Processing Symposium, 1998.

[5] Khaled M. Hammouda and Mohamed S. Kamel, "Efficient phrase-based document Indexing for web document clustering", IEEE tran. on knowledge and data engineering, Vol.16, No.10, pp.1279-1296, Oct., 2004.

[6] M. A. Hernandez and S. J. Stolfo, "The merge/purge problem for large databases", Proceedings of the ACM SIGMOD International Conference on Management of Data, pp.127-138, May, 1998.

[7] P. S. Bradley, Uama M Fayyad, "Refining Initial Points for K-Means Clustering", Proceedings of the Fifteenth International Conference on Machine Learning, 1998.

[8] W. Klosgen and J.M. Zytkow, 'Handbook of Data Mining and Knowledge Discovery', Oxford University Press, New York, 2002.

[9] W. L. Low, M. L. Lee, and T. W. Ling, "A knowledge-based approach for duplicate elimination in data cleaning", Information Systems, Vol.26, No.8, pp.585-606, Dec., 2001.

[10] O. Zamir and O. Etzioni, "Fast and intuitive clustering of web documents", KDD97, pp.287-290, 1997.

[11] P. Soucy and G. W. Mineau, "A Simple KNN Algorithm for Text Categorization", Proceeding of 1st. IEEE international conference on data mining, Vol.28, pp.647-648, 2001.

[12] Ying Zhao and George Karypis, "Web clustering: Evaluation of hierarchical clustering algorithms for document datasets", Proceedings of the eleventh international conference on Information and knowledge management, pp.515-524, Nov., 2002.

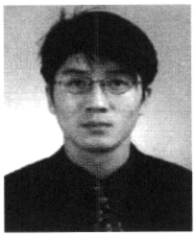
[13] Y. Wang, "Use link-based clustering to improve web search results", Proceedings of 2nd. conference on web information systems engineering, Vol.1, pp.115-123, Dec., 2001.

[14] 강동혁, 주길홍, 이원석, "대용량 문서 데이터베이스를 위한 효율적인 점진적 문서 클러스터링 기법", 정보처리학회논문지 D, 제10-D권, 제1호, pp.57-66, 2003. 02.

[15] 강승식, "HAM : 한국어 분석 모듈", http://nlp.kookmin.ac.kr.

[16] 김제욱, 김한준, 이상구, "베이지언 문서분류시스템을 위한 능동적 학습 기반의 학습문서집합 구성방법", 정보과학회논문지: 소프트웨어 및 응용, 29권, 12호, pp.966-978, 2004. 09.

[17] 박우창 외, '데이터 마이닝:개념 및 기법', 자유아카데미, 2003. 09.



고 석 범

e-mail : sbko73@yahoo.co.kr
1996년 동서대학교 컴퓨터공학(학사)
1998년 아시카가 공대 정보시스템공학(석사)
2001년~2004년 육군사관학교 전자계산학과 전임강사
현 재 부경대학교 전자계산학 박사과정
관심분야 : 분산데이터베이스, 텍스트마이닝, 문서 클러스터링, XML

윤 성 대

e-mail : sdyoun@pknu.ac.kr
1980년 경북대학교 컴퓨터공학(학사)
1984년 영남대학교 전자계산학(석사)
1997년 부산대학교 전자계산학(박사)
1986년~현재 부경대학교 전자계산학과 교수
관심분야 : 병렬운영체제, 데이터마이닝, Multi-threaded architecture