경영정보학연구
제12권 제2호
2002년 6월

사례기반 추론을 이용한 한글 문서분류 시스템*

이 재 식**, 이 종 운***

A Hangul Document Classification System using Case-based Reasoning

Jae Sik Lee**, Jong Woon Lee***

In this research, we developed an efficient Hangul document classification system for text mining. We mean 'efficient' by maintaining an acceptable classification performance while taking shorter computing time. In our system, given a query document, k documents are first retrieved from the document case base using the k-nearest neighbor technique, which is the main algorithm of case-based reasoning. Then, TFIDF method, which is the traditional vector model in information retrieval technique, is applied to the query document and the k retrieved documents to classify the query document. We call this procedure 'CB_TFIDF' method. The result of our research showed that the classification accuracy of CB_TFIDF was similar to that of traditional TFIDF method. However, the average time for classifying one document decreased remarkably.

* 이 논문은 2000년도 두뇌한국 21 사업에 의하여 지원되었음.
** 아주대학교 경영대학
*** 대우정보시스템 건설시스템팀

Ⅰ. 서 론

전 세계의 데이터의 양은 급격히 증가하고 있으며, 오늘날의 기업, 정부, 그리고 학계에는 인터넷으로부터 쏟아지는 반구조적인 혹은 텍스트(Text) 형태의 데이터의 홍수로 넘쳐 나고 있다. 데이터 마이닝(Data Mining)은 방대한 양의 데이터를 분석하여 그 속에서 의미 있고 이해할 수 있는 패턴(Pattern), 즉 지식(Knowledge)을 찾아내는 분야로서, 특별히 텍스트 형태의 데이터를 분석하는 것을 텍스트 마이닝(Text Mining)이라고 한다[Trybula, 1999; Berry and Linoff, 2000]. 일반적으로 텍스트를 기반으로 작성된 문서는 단순한 수치적(Numerical) 데이터에 비해 더 풍부한 정보를 포함하고 있다. 따라서 텍스트 마이닝은 기존의 수치적 데이터 마이닝보다 더 유용한 지식을 추출할 수 있다.

기계 학습(Machine Learning) 분야인 사례기반 추론(Case-based Reasoning 혹은 Memory-based Reasoning)이나 인공신경망(Artificial Neural Network)을 이용한 문서분류 시스템(Document Classification System) 또는 문서여과 시스템(Document Filtering System)에 대한 기존 연구는 많이 있으며[Lewis and Ringuette, 1994; Gudivada et al., 1997; Joachims, 1997; Cho, 1999; 김시천, 1999; 이형일, 1999; 안수산과 신경식, 2000], 그 성능 또한 뛰어난 것으로 입증된 것이 많다. 하지만, 대부분의 연구가 영어 문서의 처리에 대한 것이며 한글 문서에 관한 연구는 드물다. 또한 대부분의 연구가 문서의 분류나 여과(이하 분류로 통칭함)의 적중률을 높이는데 초점을 둔 것들이다. 텍스트 데이터는 데이터의 성격상 문서의 개수가 늘어남에 따라 데이터의 처리에 소요되는 시간이 기하급수적으로 증가하게 된다. 그러므로, 문서분류 시스템은 '문서를 얼마나 정확하게 분류하느냐'를 평가하는 '시스템의 분류 적중률' 못지 않게 '문서를 얼마나 빠른 시간 내에 만족할만한 수준으로 분류하느냐'를 평가하는 '시스템의 구현가능성 또는 효율성'이 중요하다고 할 수 있다.

본 연구에서는 많은 양의 한글 문서를 만족할 만한 성능을 유지하면서 빠른 시간 내에 분류할 수 있는 기법을 개발하고자 한다. 이를 위해 본 연구에서는 사례기반추론 기법과 전통적인 정보검색(Information Retrieval) 기법인 TFIDF (Term Frequency Inverse Document Frequency) 방법을 결합하여, 새로운 문서분류 기법을 제시한다.

본 논문은 다음과 같이 구성되어 있다. 제 2절에서는 본 연구의 이론적 토대가 되는 정보 검색과 문서분류에 관한 기존 연구들을 간략히 살펴본다. 제 3절에서는 본 연구에서 개발한 한글 문서분류 시스템의 전체적인 구조를 살펴보고, 시스템의 각 구성요소들의 세부적인 특징들을 설명한다. 제 4절에서는 본 연구에서 제안하는 새로운 문서분류 기법에 대해서 설명하고, 제 5절에서는 이 새로운 기법과 기존 문서분류 기법의 성능을 비교 평가한다. 마지막으로 제 6절에서는 본 연구의 결론 및 향후 연구과제를 제시한다.

Ⅱ. 정보 검색

2.1 정보 검색 시스템에 대한 기존 연구

정보 검색은 문서의 집합으로부터 자연어로 표현된 사용자의 정보요구사항(Information Needs)을 만족시키는 문서의 검색을 연구하는 분야로서, 검색이 효과적·효율적으로 수행되기 위한 정보의 표현·저장·조직화 그리고 이에 대한 접근을 다룬다[Baeza-Yates and Ribeiro-Neto, 1999]. 사용자의 정보요구사항은 사용자의 관심과 일치하는 정보에 대한 검색이 용이하도록 표현되고 조직화되어야 하는데, 사용자의 정보요구사항을 문장으로 적절하게 특징짓는 것은 쉬운 일이 아니다. 그러므로, 정보검색이 효과적·효율적으로 수행되도록 하기 위해서는 우선 정보요구사항을 검색 엔진이 처리할 수 있는 간결

한 질의어로 바꿔주어야 한다. 가장 일반적인 질의어의 형태가 키워드(Keyword)들 혹은 색인어들의 집합이다.

일반적인 정보 검색 시스템이 채용하고 있는 모델은 문서의 내용을 구성하는 단어들에 중요도를 나타내는 가중치를 부여하고, 이를 근거로 질의문서(Query Document) 또는 질의어와 검색문서 간의 유사도를 측정하는 방법을 사용하고 있다. 가장 기본적인 모델로는 부울리안(Boolean) 모델[Gudivada et al., 1999]과 벡터(Vector) 모델[Baeza-Yates and Ribeiro-Neto, 1999]이 있는데, 본 연구에서는 벡터 모델에서 일반적으로 사용하는 TFIDF 방법을 사용하였다.

벡터 모델로 표현된 문서에서는 문서에 포함된 단어의 개수가 곧 문서의 속성 개수가 되는데, 이 속성의 개수를 줄이기 위해 기본적으로 불용어(Stopword) 제거 및 스테밍(Stemming) 방법이 사용된다. 불용어는 한글문서에서의 '-은', '-는', '-이', '-가', '-이다' 따위의 조사나 어미, 영어문서에서의 'a', 'the', 'of'와 같이 색인어로서의 가치가 없는 단어를 말한다. 스테밍은 한글문서에서 '간다', '가는'이 '가다'로 대체되고 영어문서에서는 'works'나 'working'이 'work'로 대체되는 것처럼 단어의 어근을 취하는 것을 말한다. 이러한 방법 외에, 단어빈도 혹은 문서빈도의 하한값을 설정해 그 개수가 극히 적은 단어들을 제거하거나[Joachims, 1997], 단어의 정보이득(Information Gain)과 같은 점수부여 방법을 통해 상위에 존재하는 단어들을 선택하는 방법[이형일, 1999]으로 문서의 속성 개수를 줄이는 방법도 연구되었다. 하지만, 문서의 양이 많은 경우에는 이러한 방법도 문서의 분류시간을 대폭 줄이지는 못하였다. 의사결정 나무[Lewis and Ringuette, 1994]나 인공신경망[Weiner et al., 1995; 안수산과 신경식, 2000] 등의 인공지능 기법을 이용한 문서분류 방법도 연구되었다.

의사결정 나무나 인공신경망 기법은 사전학습기법(Eager Learning Technique)들이므로 문서를 분류하기 위한 모델을 미리 구축해 놓은 후에 문서분류 작업을 수행하여야 한다. 의사결정 나무는 가지치기를 하기 위한 변수, 즉 문서분류의 경우에는 단어를 선정하여 가지치기를 하면서 나무를 구축해 나간다. 하지만, 만일 의사결정 나무의 첫 번째 가지치기에 사용된 단어가 새로운 질의문서에 포함되어 있지 않다면, 질의문서에 대한 분류 작업은 시작조차 못하게 될 것이다. 또한 새로운 분류 유형이 생기면 그 때마다 새로운 의사결정 나무를 구축하여야 한다.

인공신경망은 모든 입력변수가 수치형이어야만 한다. 문서분류는 단어들을 다루는 것이기 때문에 이 단어들을 수치화 시켜야만 인공신경망을 사용할 수 있는 것이다. 안수산과 신경식[2000]의 연구에서는 수신된 영문 e-Mail이 스팸(Spam) 메일인지 아닌지를 구분하는 이진분류(Binary Classification) 문제를 다루었다. 하지만, 본 연구에서처럼 문서를 여러 유형으로 분류하는 문제인 경우에는 인공신경망의 적용이 거의 불가능하다. 그들의 연구에서처럼 스팸 메일인지 아닌지를 분류하는 이진분류의 경우에는 스팸 메일이 보유하는 단어 또는 스팸이 아닌 메일이 보유하는 단어들을 입력변수로 사용하여 인공신경망을 구축할 수 있지만, 여러 유형으로 문서를 분류할 수 있는 인공신경망은 입력변수 자체를 선정하는 것이 거의 불가능하기 때문이다. 또한 인공신경망을 구축하였다 하더라도, 만일 새로운 분류 유형이 생기면 그 때마다 새로운 인공신경망을 구축하여야 한다.

이와 같은 의사결정 나무와 인공신경망의 단점을 사례기반 추론에서는 극복할 수 있다. 우선, 사례기반 추론은 사후학습기법(Lazy Learning Technique)이기 때문에 사전에 모든 유형으로 분류할 수 있는 모델을 구축해 놓지 않아도 된다. 새로운 유형이 생겼을 때에 이를 사례베이스에 추가함으로써 학습이 되는 것이다. 또한 사례기반 추론은 수치형과 범주형 변수를 모두 자연스럽게 다룰 수 있으며, 부분일치(Partial Ma-

tching)에 의하여 해답을 주기 때문에 다소 사례베이스의 지식이 부족하더라도 질의문서에 대해서 가장 근접한 유형을 분류 결과로 제시할 수 있다.

본 연구에서 사용하는 사례기반 추론(혹은 메모리기반 추론) 기법을 사용한 문서분류에 대한 연구[김시천, 1999; 이형일, 1999]도 수행되었다. 김시천은 질의문서에서 색인어로 사용될 단어들을 모두 찾아내고, 이것들과 사례베이스에 저장된 문서가 포함하고 있는 모든 단어들과 비교하여 유사한 문서를 검색하였다. 그가 사용한 방법이 분류에 얼마나 시간이 걸렸는지가 그의 연구결과에 제시되어 있지는 않지만, 그의 분류 방법은 시간이 매우 오래 걸릴 것이라는 것을 쉽게 짐작할 수 있다. 하지만, 우리의 연구에서 개발된 방법은 질의문서에 포함된 단어들의 일부만을 사용하고, 사례베이스에 저장된 문서들도 일부만을 사용하므로 분류에 사용되는 시간이 대폭 감소하였다.

즉, 본 연구에서는 분류할 질의문서와 비교할 문서들을 검색할 때에, 사례기반 추론의 k-최근접이웃(k-Nearest Neighbor) 방법을 이용하여 사례베이스로부터 k개의 문서를 검색한 후에, 이 문서들만을 대상으로 TFIDF를 수행함으로써 문서분류 시스템의 효율성을 향상시키는 방법을 제안한다.

2.2 정보 검색을 위한 벡터 모델

본 절에서는 본 연구에서 채용한 벡터 모델인 TFIDF 기법[Baeza-Yates and Ribeiro-Neto, 1999]에 대해서 간략히 소개한다. 벡터 모델은 부울리안 모델의 0 또는 1의 가중치의 한계를 극복하고 질의문서와 검색문서 간의 부분일치를 가능하게 한다. 즉, 질의문서와 검색문서의 단어들에 연속형 수치의 가중치를 부여하고, 이 가중치들을 이용하여 유사도를 계산한 후, 상위의 유사도를 갖는 문서들을 검색해오는 방법인데, 부울리안 모델에 의한 방법보다 정확하게 사용자의 정보요구사항에 부합하는 문서들을 검색할 수 있다는 장점 때문에 현재 널리 쓰이고 있다[Mladenic, 1999].

벡터 모델은 하나의 문서를 t개의 정규화된 단어로 구성된 t-차원의 벡터로 표현한다. 즉,

w_{ij} : 검색문서 j의 단어 i의 가중치 ($w_{ij} \geq 0$).

w_{iq} : 질의문서 또는 질의어 q의 단어 i의 가중치 ($w_{iq} \geq 0$).

t : 검색문서와 질의문서 내의 단어들의 개수

일 때, 질의문서의 벡터 $\vec{q}$는 식 (2-1)과 같이 나타낼 수 있다.

$$\vec{q} = (w_{1q}, w_{2q}, \cdots, w_{tq}) \quad (2\text{-}1)$$

또한, 검색문서의 벡터 $\vec{d_j}$는 식 (2-2)와 같이 나타낼 수 있다.

$$\vec{d_j} = (w_{1j}, w_{2j}, \cdots, w_{tj}) \quad (2\text{-}2)$$

벡터 모델에서 유사도 산출 공식은 식 (2-3)과 같다.

$$sim(d_j, q) = \frac{\vec{d_j} \cdot \vec{q}}{|\vec{d_j}| \times |\vec{q}|} = \frac{\sum_{i=1}^{t} w_{ij} \times w_{iq}}{\sqrt{\sum_{i=1}^{t} w_{ij}^2} \times \sqrt{\sum_{i=1}^{t} w_{iq}^2}} \quad (2\text{-}3)$$

두 문서간의 유사도를 산출하기 위해서는 먼저 각 문서가 보유하고 있는 단어들의 가중치를 계산해야 한다.

검색문서 j내의 단어 i의 가중치 w_{ij}와 질의문서 q내의 단어 i의 가중치 w_{iq}는 일반적으로 TFIDF 방법으로 계산되는데, TFIDF란

TF(Term Frequency)와 IDF(Inverse Document Frequency)를 곱한다는 의미이다. TF, 즉 단어빈도는 문서에서 단어가 나타나는 빈도를 의미하며 식 (2-4)와 같이 계산된다.

$$TF_{ij} = \frac{f_{ij}}{\max f_j} \quad (2\text{-}4)$$

여기서,

f_{ij}: 검색문서 j 내의 단어 i의 출현 회수

$\max f_j$: 검색문서 j에서 가장 많이 출현한 단어의 출현 회수

예를 들어, 제 3번 검색문서에서 가장 많이 출현한 단어인 '농사'가 25번 출현하였고, 그 문서에서 '벼'라는 단어가 10번 출현하였다면, 제 3번 검색문서에서 '벼'의 TF 값은 10/25, 즉 0.4가 된다.

IDF를 구하기 전에, 먼저 DF(Document Frequency)를 계산해야 하는데, DF, 즉 문서빈도는 보유한 전체 문서 중 해당 단어를 갖고있는 문서의 비율을 나타내는 것으로서 식 (2-5)와 같이 계산된다.

$$DF_i = \frac{n_i}{N} \quad (2\text{-}5)$$

여기서 n_i: 단어 i를 가진 문서의 개수

N: 보유한 전체 문서의 개수

예를 들어, 우리가 보유한 문서가 모두 1,000개라고 하고, '벼'라는 단어를 포함한 문서가 모두 550개라고 하면, '벼'의 DF 값은 550/1000, 즉 0.55가 된다. TF의 경우에는 그 값이 크면, 해당 단어가 그 문서에서 중요도가 높다고 할 수 있지만, DF의 값이 크면, 해당 단어가 여러 문서에 걸쳐 나타나므로 어떤 특정 문서를 분류할 때에 사용할 수 있는 단어로서의 가치가 떨어진다고 할 수 있다. 그러므로 DF의 역수에 log를 취한 IDF를 사용한다.

IDF, 즉 역문서빈도는 식 (2-6)과 같이 계산되는데, 적은 개수의 문서에 걸쳐 나타난 단어의 가중치는 높이고, 많은 개수의 문서에 걸쳐 나타난 단어의 가중치는 낮추는 효과를 준다.

$$IDF_i = \log \frac{N}{n_i} \quad (2\text{-}6)$$

우리의 예인 '벼'의 경우에 IDF 값은 log(1000/550), 즉 0.2596이 된다.

위의 식 (2-4)와 식 (2-6)을 사용하여, 검색문서 j 내의 단어 i의 가중치, 즉 w_{ij}는 식 (2-7)과 같이 계산된다.

$$w_{ij} = TF_{ij} \times IDF_i \quad (2\text{-}7)$$

그러므로, 제 3번 검색문서에서 단어 '벼'의 가중치는 $0.4 \times 0.2596 = 0.1038$이 된다.

질의문서 q내의 단어 i의 가중치 w_{iq}도 w_{ij}를 구할 때와 마찬가지로 식 (2-8)과 같이 계산하는데, Salton and Buckley[1988]는 단어에 적절한 가중치를 부여하는 다양한 연구를 한 결과, w_{iq}를 구하기 위한 TF는 식 (2-9)를 사용하여 계산할 것을 제안하였다.

$$w_{iq} = TF_{iq} \times IDF_i \quad (2\text{-}8)$$

$$TF_{iq} = \left(0.5 + \frac{0.5 f_{iq}}{\max f_q}\right) \quad (2\text{-}9)$$

여기서, f_{iq}: 질의문서 q내의 단어 i의 출현 회수

$\max f_q$: 질의문서 q에서 가장 많이 출현한 단어의 출현 회수

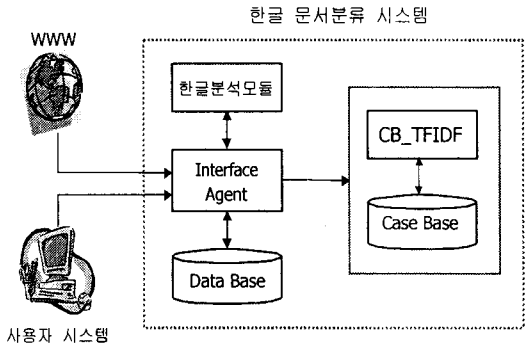
w_{ij}를 구할 때의 TF는 0 과 1 사이의 값을 갖지만, w_{iq}를 구할 때의 TF는 식 (2-9)에 의하여 0.5와 1 사이의 값을 갖게 된다. 즉, 질의문서에서는 단어간의 가중치의 차이가 작아지는 효과가 있다.

TFIDF는 간략히 설명하면, 각 문서에 존재하

는 단어에 대해 가중치를 부여함으로써 문서의 특징을 표현한 후에, 단어들의 가중치의 벡터로 표현되는 두 문서간의 유사도 비교를 가능하게 하는 방법이다. 그러나 문서가 보유한 모든 단어들의 가중치 벡터로 문서들을 비교한다는 것은 문서의 크기가 커지거나 문서의 개수가 많아질수록 비교에 소요되는 시간이 길어진다는 단점이 있다.

Ⅲ. 한글 문서분류 시스템 CB_HDC의 구조

본 연구에서 구현된 한글 문서분류 시스템인 CB_HDC(Case-based Hangul Document Classifier)의 전체적인 구조는 <그림 3-1>과 같다. 한글 색인어의 추출 작업을 위해서는 한성대학교 한글공학연구소가 제작한 한글분석 모듈인 HAM (Hangul Analysis Module)을 사용하였으며[한글공학연구소, 1999], 문서분류 엔진은 사례기반 추론 시스템으로 구현되었다.



<그림 3-1> 한글 문서분류 시스템의 구조

3.1 인터페이스 에이전트

인터페이스 에이전트(Interface Agent)는 사용자 컴퓨터 내의 문서, 즉 로컬(Local) 문서뿐만 아니라 웹 상의 HTML 문서도 수집할 수 있도록 설계되었다. 우선 로컬 문서의 경우에는 사용자가 자신의 컴퓨터 내에 문서가 저장되어 있는 디렉토리(Directory)를 지정하면 인터페이스 에이전트가 한글 분석모듈을 통해 문서의 색인어들을 추출해 데이터베이스에 저장한다. 웹 상의 문서의 경우에는 사용자가 인터페이스 에이전트에서 검색엔진을 선택해 질의어를 입력하면, 인터페이스 에이전트가 검색된 문서들의 색인어들을 로컬 문서와 마찬가지의 방식으로 추출해 데이터베이스에 저장한다. 이미 분류가 되어 있는 문서들로부터 추출된 색인어들은 사례베이스에 저장되어 있다. 분류가 아직 안된 문서로부터 추출된 색인어들은 분류용 질의문서로 구성되어 문서분류엔진에 입력되어 문서분류 시스템을 구동한다.

3.2 한글분석 모듈

색인어 추출을 위한 불용어의 정의나 스테밍 방법에 대한 연구가 많이 되어있는 영어 문서와는 달리, 한글 문서는 그 색인어의 추출 방법이 상대적으로 까다롭고 이에 대한 연구도 많이 존재하지 않는 실정이다. 특히 정보 검색과 관련해 문제가 되는 것은 복합명사와 고유명사 혹은 신조어의 존재인데, 띄어쓰기에 대한 명확한 규정이 없는 한글의 복합명사는 띄어쓰기의 방법에 따라 추출되는 색인어의 형태가 달라지게 된다. 본 연구에서는 한글 문서분류 시스템의 구현상의 부담을 줄이고자 이미 개발되어 있는 한글분석 모듈인 HAM을 CB_HDC의 하나의 구성요소로 사용하였다. HAM은 C 언어로 제작된 쉐어웨어(Shareware)로서 한글 문서의 형태소 분석과 자체 불용어 사전 참조를 통한 색인어 추출을 수행한다.

3.3 사례베이스

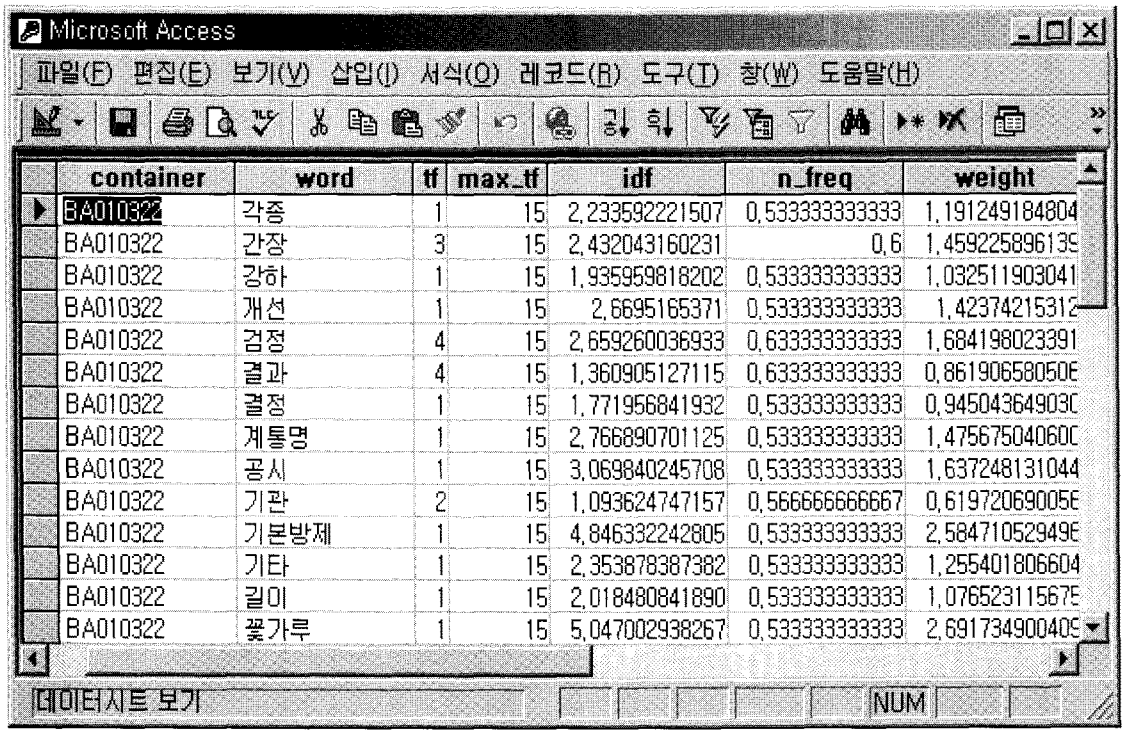
사례베이스에는 <그림 3-2>와 같이 이미 분류

번호가 부여된 문서들의 속성들이 저장되어 있다. 속성으로는 각 문서가 포함하고 있는 모든 단어들의 문서 내 출현 횟수, 문서 내에서 가장 많이 출현한 단어의 출현 횟수, 역문서빈도, 단어의 출현 횟수를 정규화 시킨 단어빈도, 그리고 이들을 이용해 계산된 단어의 가중치 등이 있다. 이 속성들은 전통적인 TFIDF 기법에 의해 계산된 것들이다. 이는, 본 연구에서 개발한 CB_TFIDF 기법이 기본적으로 TFIDF 기법을 사용하고 있기 때문이다. 그러므로, 사례기반 시스템을 구축할 때에 수행되기도 하는 속성선정에 대한 고려는 별도로 하지 않았다.

CB_HDC의 사례베이스는 사용자가 불용어들을 추가하거나 문서빈도의 상·하한 값을 재설정 할 때마다 다시 구성되어 사례베이스의 크기를 크게 줄일 수 있도록 설계되었다. 즉, HAM이 기본적으로 제거하는 불용어 이외에, 사용자가 임의의 단어를 불용어로 등록하여 문서의 속성 개수를 줄일 수 있으며, 사례베이스 전체 문서에 걸쳐 너무 많이 등장하여 색인어로서의 가치가 떨어지는 단어와 극히 적게 등장하는 단어는 문서빈도의 상·하한 값의 재설정을 통해 제거할 수 있는 것이다. 그러나, 본 연구에서는 사례베이스 문서에 인위적인 축소를 가하지 않은 상태에서 CB_HDC의 분류 성능을 측정하고자 했기 때문에 사례베이스의 크기를 항상 최대 상태로 유지한 상태에서 연구를 수행하였다.

3.4 문서분류 엔진

CB_HDC의 문서분류 엔진은 사례기반 추론으로 작동한다. 사례기반 추론이란 인공지능의 기계학습 기법들 중의 하나로서 시스템 기억장치에 과거에 이미 해결된 사례들을 저장하여 놓



container	word	tf	max_tf	idf	n_freq	weight
BA010322	각종	1	15	2,233592221507	0,533333333333	1,191249184804
BA010322	간장	3	15	2,432043160231	0,6	1,459225896139
BA010322	강하	1	15	1,935959818202	0,533333333333	1,032511903041
BA010322	개선	1	15	2,6695165371	0,533333333333	1,42374215312
BA010322	검정	4	15	2,659260036933	0,633333333333	1,684198023391
BA010322	결과	4	15	1,360905127115	0,633333333333	0,861906580505
BA010322	결정	1	15	1,771956841932	0,533333333333	0,945043649030
BA010322	계통명	1	15	2,766890701125	0,533333333333	1,475675040600
BA010322	공시	1	15	3,069840245708	0,533333333333	1,637248131044
BA010322	기관	2	15	1,093624747157	0,566666666667	0,619720690056
BA010322	기본방제	1	15	4,846332242805	0,533333333333	2,584710529496
BA010322	기타	1	15	2,353878387382	0,533333333333	1,255401806604
BA010322	길이	1	15	2,018480841890	0,533333333333	1,076523115675
BA010322	꽃가루	1	15	5,047002938267	0,533333333333	2,691734900405

<그림 3-2> 사례베이스의 구조

고, 현재에 발생한 새로운 사례를 해결할 때에 기억장치에 저장된 사례들 중 가장 유사한 사례를 찾아 그 사례를 현재 사례에 맞게끔 조정하여 추천하는 것이다[Kolodner, 1993]. 특히 사례기반 추론 시스템에서는 지식을 사례의 형태로 저장하기 때문에 기존의 인공지능 기법의 문제점으로 지적 되어온 지식 획득 병목현상의 문제를 완화할 수 있다는 장점이 있다. Aamodt and Plaza[1996]는 <그림 3-3>과 같이 CBR 과정을 4R이라고 부르는 4 단계로 나누어 보았다.

각 단계에 대한 간략한 설명은 다음과 같다.

1) 검색(Retrieve): 새롭게 입력된 질의 사례와 가장 유사한 과거의 사례를 찾는다.
2) 재사용(Reuse): 질의 사례와 검색된 사례 간의 차이를 고려하여 해를 제공한다.
3) 수정(Revise): 재사용 단계에서 제공된 해가 질의 사례의 해로서 적합하지 않았을 때에 해를 수정한다.
4) 유지(Retain): 질의 사례에 대해 제안된 해를 지식으로 유지한다.

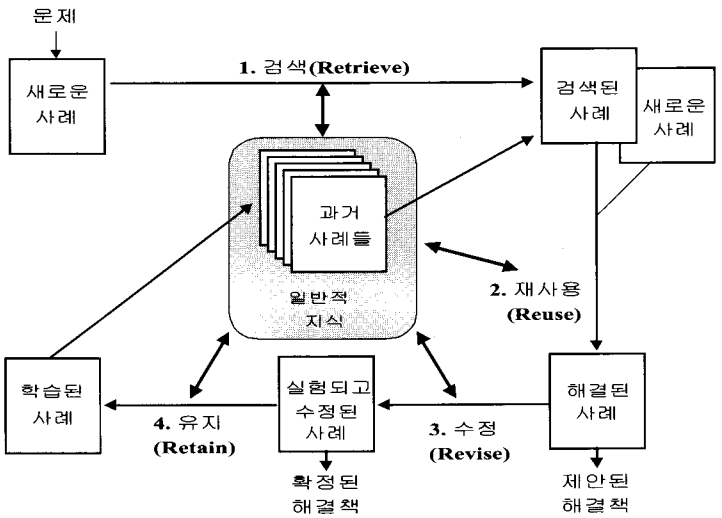
대부분의 인공지능 기법들은 훈련용 사례들로부터 추출한 일반화된 지식을 사용하여 문제를 해결하기 때문에 특별한 상황에 적용되는 해결책을 제시하기 어렵다. 반면에 사례기반 추론 시스템은 과거의 사례들을 바탕으로 문제를 해결하고 새로운 사례들을 계속해서 사례베이스에 축적해 나감으로써 지식의 불완전을 극복할 수 있다[Watson, 1997].

분류하고자 하는 문서가 입력되면, 인터페이스 에이전트가 색인어들을 추출하고 이것들에게 가중치를 부여하여 질의문서를 구성한다. 이 질의문서가 문서분류 엔진에 입력되면, 문서분류 엔진은 사례베이스로부터 k개의 문서를 검색한 후, 질의문서와 각 검색문서들 간의 유사도를 계산해서 유사도 점수가 가장 높은 검색문서가 갖는 분류범주(Category)를 입력 문서에 할당한다. 상세한 과정은 제 4.3절에서 설명한다.

Ⅳ. 사례기반 추론 문서분류 기법 CB_TFIDF

4.1 사용 데이터

본 연구에서는 농업 진흥청 산하 사단법인 농진회에서 제작한 '농업기술 CD-ROM'에 수록된 문서들을 데이터로 사용하였다. 이 CD-ROM은



<그림 3-3> Aamodt and Plaza의 사례기반 추론 과정

모두 3장으로 구성되어 있는데 각각의 내용은 <표 4-1>과 같다. <표 4-1>에서 '농업환경'이나 '농기계' 같이 여러 CD에 중복되어 등장하는 내용은 해당 CD가 다루는 분야에만 해당하는 것으로서 동일한 내용들이 아니다.

<표 4-1> 농업기술 CD-ROM의 내용

CD 번호	내 용
CD 1	식량작물, 특용작물, 잠업, 농업환경, 농기계, 농업경영, 농산물이용, 생활과학, 농기자재
CD 2	채소, 과수, 화훼, 농업환경, 농기계, 농업경영, 농산물이용, 농기자재
CD 3	축산, 가축위생, 농기계, 농업경영, 농업환경, 농기자재, 농산물이용

모든 문서들은 <그림 4-1>과 같이 HTML로 작성되어 있다. 각 문서는 8자리로 된 고유한 번호를 가지고 있는데, 이중 앞의 6자리 문자와 숫자는 <표 4-2>와 같이 문서가 속한 분류범주를 의미하며, 마지막 2자리 숫자는 해당 분류범주에서 문서를 식별할 수 있는 번호에 해당한다

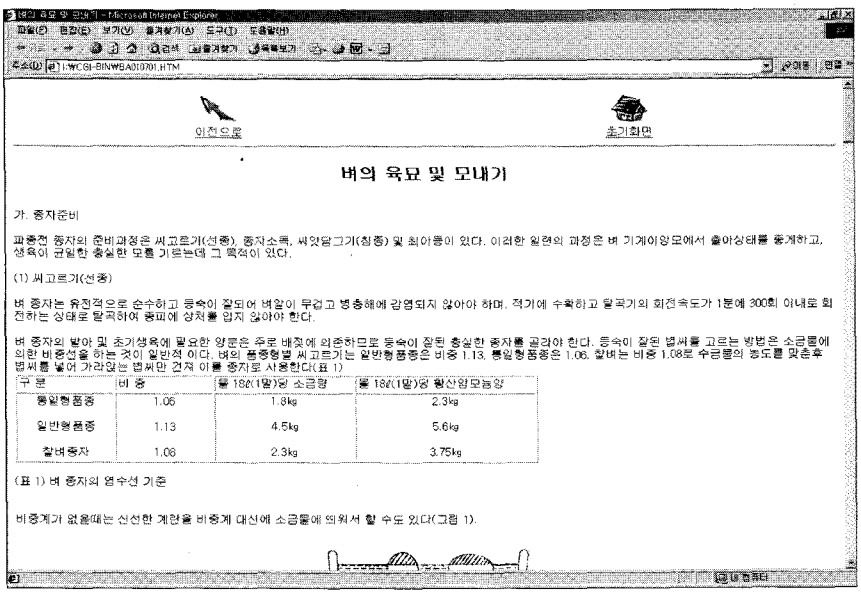
<표 4-2> 문서번호 BA010702의 의미

분류기호	분류범주	분류단계
BA	식량작물	대 분 류
01	벼	중 분 류
07	육묘	소 분 류
02	BA0107의 2번째 문서	

본 연구에서는 제 1번 CD-ROM에 수록된 문서들을 사용하여 사례베이스를 구성하고, 훈련용 문서와 검증용 문서를 준비하였다. 전체 1,954개의 문서들 중에서 포함된 문서의 개수가 너무 적은 분류범주들을 제거하기 위해 대분류 기준으로 문서의 개수가 100개 미만인 문서들은 연구 데이터에서 제외하였다. 그 결과 총 1,750개의 문서가 사용되었는데, 그 구성은 <표 4-3>

<표 4-3> 연구에 사용된 문서의 구성비

	문서개수	비율(%)	레코드 수
사례베이스용 문서	1400	80	277,053
훈련용 문서	175	10	29,784
검증용 문서	175	10	30,677



<그림 4-1> 연구에 사용된 문서의 예

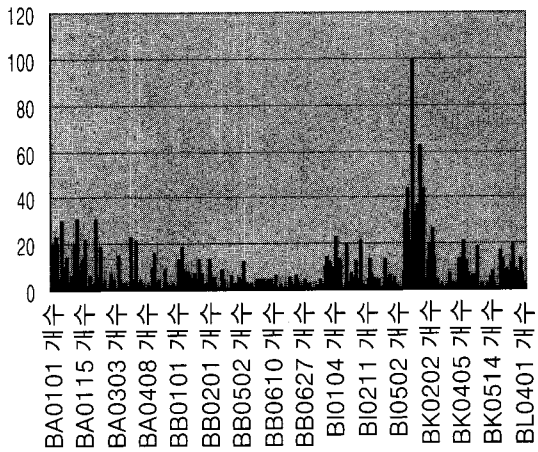
과 같다. <표 4-3>에서 레코드 수는 해당 문서집합 내 문서가 보유한 단어의 개수를 모두 합한 것이며 이는 해당 문서집합의 크기를 의미한다.

훈련용 문서와 검증용 문서를 통칭할 때에는 '질의문서'라는 용어를 사용하기로 한다. 본 연구에서 사용된 문서의 분류범주의 개수는 <표 4-4>와 같다.

<표 4-4> 연구에 사용된 문서의 분류범주의 개수

	대분류	중분류	소분류
전체 문서	5	28	212
사례베이스용 문서	5	28	199
훈련용 문서	5	23	88
검증용 문서	5	26	91

문서분류 시스템의 분류 적중률의 측정은 질의문서에 대해서 문서분류 시스템이 찾아낸 문서의 분류범주가 질의문서의 분류범주와 얼마나 일치하는가를 판단함으로써 이루어진다. <그림 4-2>는 본 연구에 사용된 전체 1,750개의 문서들의 소분류 기준에 의한 분포를 보여주고 있는데, 많은 분류범주에 걸쳐 문서의 개수가 10개 미만인 것이 많다.

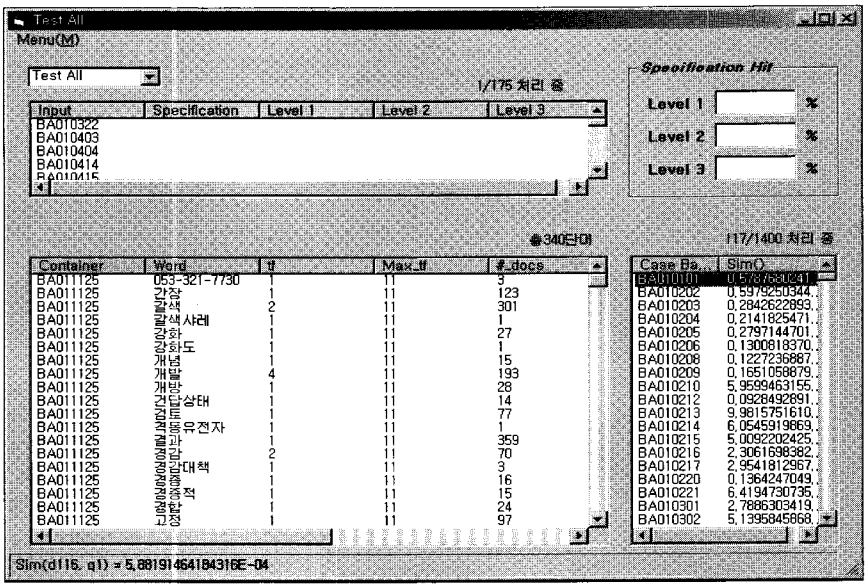


<그림 4-2> 소분류 기준의 문서 분포

이렇게 소분류 기준으로 본 문서의 불균형적인 분포는, 제 5절의 실험 결과에서도 알 수 있겠지만, 소분류의 분류 성능 저하의 요인이 되었다.

4.2 분류 방법 I : TFIDF 방법에 의한 분류

사례베이스는 TFIDF에 의해 가중치가 부여된



<그림 4-3> 전체 사례베이스에 TFIDF 방법을 적용한 문서분류 화면

모든 단어들을 보유하고 있다. 여기서 모든 단어들을 보유했다는 의미는 제 3.3절에서 언급한 바와 같이 사례베이스 내의 단어 개수를 줄이기 위한 어떠한 조작도 가하지 않았다는 의미이다. 분류 방법 Ⅰ은 각 질의문서에 대해 사례베이스 전체를 사용하여 전통적인 기법인 TFIDF에 의한 분류를 수행하는 것이다. <그림 4-3>은 전체 사례베이스 보유 문서를 사용하여 TFIDF 방법으로 훈련용 문서를 분류하는 화면이다.

<그림 4-3>에서 왼쪽 상단의 리스트는 질의문서의 목록을 표시하며, 오른쪽 하단의 리스트는 질의문서와 비교되는 사례베이스 내의 문서의 목록 및 질의문서와의 유사도를 표시하고 있다. 왼쪽 하단의 가장 큰 리스트는 질의문서와 비교되고 있는 사례베이스 문서의 속성 값들을 보여준다. <그림 4-3>의 화면에서 하나의 질의문서에 대한 모든 사례베이스 문서와의 비교가 끝나면 질의문서 목록에 사례베이스에서 가장 유사도가 높은 문서가 추천된다. 그러면, 추천된 문서와 질의문서의 번호가 나타내는 분류범주를 비교하여 대분류, 중분류, 소분류의 일치 여부를 표시하게 된다. 질의문서 목록의 모든 문서에 대한 분류가 끝나면 오른쪽 상단의 텍스트 상자에 적중률이 표시된다.

4.3 분류 방법 Ⅱ : CB_TFIDF 방법에 의한 분류

본 연구에서는 좀더 효율적인 검색 및 분류를 수행하기 위해서 CB_TFIDF(Case-Based TFIDF)라는 새로운 방법을 개발하였다. CB_TFIDF 방법은 기본적으로 사례기반 추론과 TFIDF가 결합된 방법으로서, 그 절차는 다음과 같다.

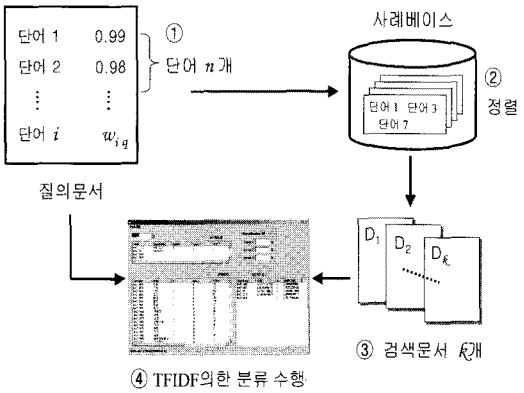
① 질의문서에서 n개의 단어를 가중치의 내림차순으로 선택한다.

② 사례베이스 내의 전체 문서에서 이 단어들 중 하나 이상을 보유한 문서들을 검색하여, 이 단어들의 보유 개수 및 보유한 단어들의 가중치의 합계를 기준으로 이 문서들을 내림차순으로 정렬한다.

③ 이들 중에서 상위에 속하는 k개의 문서를 선택한다.

④ 질의문서에 대해 선택된 k개의 문서를 사용하여 TFIDF에 의한 분류를 수행한다.

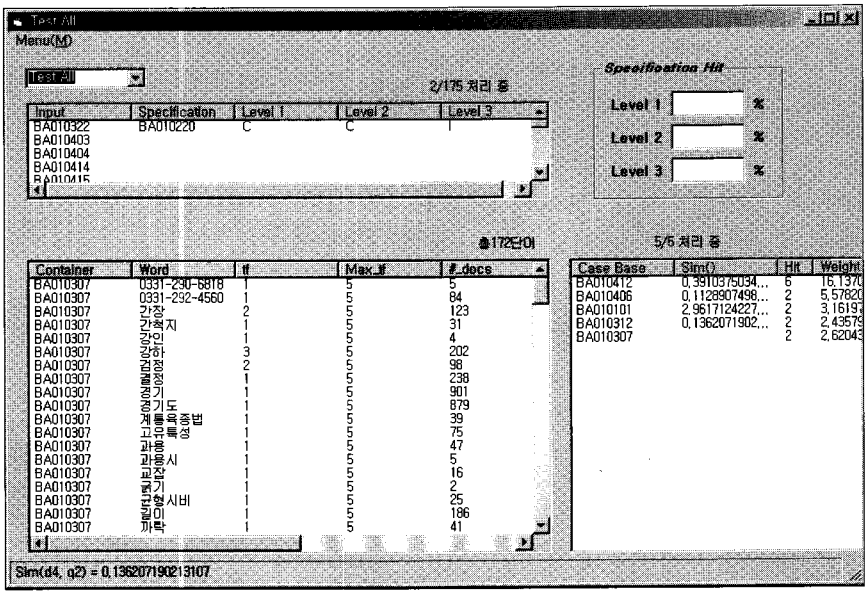
CB_TFIDF 방법의 과정을 도식화하면 <그림 4-4>와 같다. 그림에 표기된 번호는 위 절차의 각 단계의 번호이다.



<그림 4-4> CB_TFIDF의 수행 과정

위의 단계 ①에서 단어의 개수 n을 10개, 20개, 30개로 변화시키고, 각 단어 개수의 경우마다 단계 ③의 k의 숫자를 3, 5, 10으로 변화시켜서 총 9개 모델을 만들어 실험을 하였다. <그림 4-5>는 훈련용 문서에 대한 CB_TFIDF 방법의 수행 화면으로서, 각 훈련용 문서에서 상위 가중치를 갖는 20개(즉, n = 20)의 단어를 선택한 후, 사례베이스에서 이들 단어를 보유한 상위 5개(즉, k = 5)의 문서를 선택해서 문서분류를 수행하는 화면이다.

<그림 4-5>의 우측 하단에 있는 사례베이스의 문서목록을 보면, 가장 첫 번째 문서가 훈련용 문서에서 선택된 20개의 단어 중 6개를 가지고 있고 그 단어들의 가중치의 합이 가장 크다는 것을 볼 수 있다.



<그림 4-5> CB_TFIDF 방법을 적용한 문서분류 화면

V. CB_TFIDF 방법의 분류 성능

본 연구는 서론에서 언급한 바와 같이 한글문서를 만족할만한 성능을 유지하면서 빠른 시간 내에 분류할 수 있는 방법을 개발하고자 하는 것이다. 그러므로, 기존의 TFIDF 방법이 일반적인 업무에 사용되는 PC에서는 허용할 수 없을 정도로 오랜 시간이 소요된다는 것을 보여줄 필요가 있다. 하지만, 본 연구의 모든 과정을 일반 PC를 사용하여 수행할 수는 없었다. 그러므로, 본 연구에서는 일반적으로 업무에 사용되는 PC, 그리고 고속·대용량의 Server급 컴퓨터 등 두 종류의 컴퓨터를 사용하였는데, 그 사양은 <표 5-1>과 같다.

본 절에서는, 기존의 TFIDF 방법이 일반 PC에서는 허용할 수 없을 정도로 오랜 시간이 소요되지만, 본 연구에서 개발한 방법인 CB_TFIDF를 사용하면 일반 PC에서도 빠른 시간 내에 한글문서를 분류할 수 있다는 것을 보여주고자 한다.

5.1 TFIDF 방법에 의한 분류 성능

일반 PC 상에서, 사례베이스에 저장된 1,400개 문서 전체를 사용하여 훈련용 문서 175개를 분류한 결과는 <표 5-2>와 같다.

<표 5-2>에서, CPU 시간은 프로세스가 질의문서와 사례베이스 문서간의 유사도 계산 및 비교를 수행하기 위해 CPU를 이용한 시간만을 측정한 것이다. 그러므로, 문서를 읽어 들이기 위한 디스크 접근 시간 등을 감안하면 실제로 분류에 걸리는 시간은 <표 5-2>에 나타난 CPU 시간보다 훨씬 길어지게 된다. <표 5-2>에서 보듯

<표 5-1> 연구에 사용된 컴퓨터의 사양

	CPU	Memory(MB)	OS
일반 PC	Pentium-II 350*2	256	Windows NT 4.0
Server	Pentium-III 667*2	896	Windows 2000 Server

<표 5-2> 일반 PC 상에서 TFIDF 방법의 문서분류 성능

질의문서	대분류(%)	중분류(%)	소분류(%)	문서당 평균 분류시간 (CPU 시간)
훈련용 문서	99.43	89.71	61.71	2시간

이 사례베이스 내의 전체 문서와 비교하며 TFIDF 방법에 의하여 한글문서 하나를 분류하는데 걸리는 시간이 일반 PC에서 2시간이 걸린다. 즉, 이 방법은 시스템 구현의 의미가 없을 정도로 그 효율성이 떨어진다고 말할 수 있다. 그러므로, 검증용 문서에 대한 분류 실험은 수행하지 않았다.

5.2 CB_TFIDF 방법에 의한 분류 성능

CB_TFIDF 방법에서, 단어의 개수 n을 10개, 20개, 30개로 변화시키고, 각 단어 개수의 경우마다 k의 숫자를 3, 5, 10으로 변화시키면서 실험을 수행하였다. 이 실험은 수행시간을 단축시키기 위하여 Server급 컴퓨터로 수행하였는데, 그 결과는 <표 5-3>과 같다.

<표 5-3>에서 평균 적중률은 대분류, 중분류, 소분류에서의 적중률들을 평균한 것이다. 이것은 각 모델의 적중률 비교시 분류범주의 대·중·소 범위마다 서로간의 우열관계가 일관되지 않을 수 있으므로 하나의 적중률 척도를 마련하기 위해서 사용한 것이다. <표 5-3>에서 볼 수

<표 5-3> Server 상에서 CB_TFIDF 방법에 의한 문서분류 성능

n	k	질의문서	대분류(%)	중분류(%)	소분류(%)	평균적중률(%)	총분류시간 (시간 : 분 : 초)
10	3	훈 련 용	93.71	85.71	60.00	79.81	00 : 15 : 14
		검 증 용	91.43	85.14	61.14	79.24	00 : 15 : 34
	5	훈 련 용	96.00	88.00	60.00	81.33	00 : 24 : 48
		검 증 용	92.00	86.29	60.00	79.43	00 : 25 : 56
	10	훈 련 용	97.14	90.86	63.43	83.81	00 : 55 : 11
		검 증 용	93.14	89.14	62.86	81.71	00 : 62 : 56
20	3	훈 련 용	94.86	88.57	62.29	81.91	00 : 15 : 52
		검 증 용	92.57	86.86	67.43	82.29	00 : 16 : 41
	5	**훈 련 용**	**96.57**	**92.53**	**66.29**	**85.13**	**00 : 33 : 50**
		검 증 용	**92.00**	**86.86**	**64.57**	**81.14**	**00 : 36 : 11**
	10	훈 련 용	96.57	90.29	63.43	83.43	00 : 61 : 37
		검 증 용	91.43	87.43	64.57	81.14	00 : 66 : 22
30	3	훈 련 용	96.57	90.85	61.71	83.04	00 : 16 : 10
		검 증 용	95.43	89.14	65.71	83.43	00 : 17 : 00
	5	훈 련 용	96.57	91.43	62.29	83.43	00 : 32 : 28
		검 증 용	93.14	86.86	61.71	80.57	00 : 36 : 00
	10	훈 련 용	91.71	89.71	61.71	81.04	00 : 68 : 06
		검 증 용	93.71	88.57	61.71	81.33	00 : 72 : 05

<표 5-4> Server 상에서 TFIDF 방법과 CB_TFIDF 방법의 문서분류 성능 비교

분류 방법	대분류(%)	중분류(%)	소분류(%)	문서당 평균 분류시간
TFIDF	92.57	89.14	62.29	34분
CB_TFIDF	92.00	86.86	64.57	12초

있듯이, 가장 뛰어난 분류 성능을 보인 모델은 훈련용 문서를 기준으로 n = 20, k = 5일 때이었다. 그러므로, 이것을 최종 문서분류 모델로 선택하였다. 이 때의 검증용 문서의 분류 결과를 TFIDF 방법의 결과와 비교해 보면 <표 5-4>와 같다.

<표 5-4>를 보면, CB_TFIDF 방법이 중분류에서의 적중률이 조금 낮기는 하지만, 전체적인 적중률이 TFIDF 방법보다 떨어진다고 할 수는 없다. CB_TFIDF 방법의 성능은 분류시간에서 확연히 드러나는데, 문서당 분류시간이 TFIDF의 34분에서 12초로, 즉 1/170로 대폭 짧아졌다. 다시 말해서, TFIDF 방법으로 하나의 문서를 분류하는 시간에 CB_TFIDF 방법으로는 검증용 문서 175개를 거의 다 분류할 수 있다는 것이다.

일반 PC에서 CB_TFIDF 방법을 이용한 문서분류 시스템의 구현이 가능한가를 점검하기 위해 훈련용 문서와 검증용 문서를 분류하는 실험을 일반 PC 상에서 수행하였는데, 분류에 소요된 시간은 <표 5-5>와 같다.

<표 5-5> 일반 PC 상에서 CB_TFIDF 방법의 분류 시간

질의문서	총 분류시간 (시간 : 분 : 초)	문서당 평균 분류시간
훈련용 문서	1 : 03 : 29	21.77초
검증용 문서	1 : 15 : 27	25.87초

<표 5-5>를 보면, CB_TFIDF 방법은 일반 PC에서도 문서당 약 20초 내지 25초 정도가 걸린다. TFIDF 방법과의 직접적인 비교는 <표 5-2>에 나타난 훈련용 문서의 분류시간을 보면 알 수 있다. 훈련용 문서 하나를 분류하는데 TFIDF 방법은 일반 PC에서 약 2시간이 걸렸지만, CB_TFIDF 방법은 약 22초밖에 걸리지 않았다. 문서 하나를 분류하는데 약 2시간이 소요되는 문서분류 시스템을 구현하는 것은 매우 어리석은 일이다. 하지만, CB_TFIDF 방법을 이용하면 문서분류의 적중률을 만족할만하게 유지하면서도 분류시간이 짧은 효율적인 한글 문서분류 시스템을 일반 PC 상에서도 구현 할 수 있다.

Ⅵ. 결론 및 향후 연구 과제

전통적인 TFIDF 방법은 그 절차의 간결성 때문에 정보 검색이나 문서분류에 폭 넓게 활용되어왔다. 하지만, 이 방법은 저장된 문서의 양이 많아지면 그 수행시간이 급격히 증가한다는 단점을 갖고 있다. 본 연구에서는 TFIDF 방법의 단점을 극복하기 위해서 사례기반 추론 기법을 접합하여 CB_TFIDF 방법을 개발하였다. CB_TFIDF 방법에서는 질의문서의 모든 단어가 아닌 가중치가 높은 몇 개의 단어만을 선택하여 사용하며, 사례베이스에 저장된 문서들도 전체가 아닌 선택된 단어의 보유 정도가 높은 몇 개의 문서만을 검색하여 분류에 사용한다. 1,400개의 문서를 보유한 사례베이스를 가지고 175개의 문서를 분류하였을 때에, TFIDF 방법은 일반 PC상에서 한 문서당 평균 분류시간이 약 2시간이나 소요되어 시스템 구현의 의미를 전혀 찾을 수가 없었다. 반면에, CB_TFIDF 방법은 TFIDF와 거의 동일한 분류 적중률을 보이면서도 한 문서당 평균 분류시간이 약 22초밖에 소요되지 않았다. 즉, CB_TFIDF 방법은 일반 PC상에서도 구현하여 사용할 수 있는 효율성이 높은 방법이라고 할 수 있다.

향후에 좀더 보완되어야 할 사항들은 다음과 같다. 첫째, CB_TFIDF 방법에서는 문서들의 속성 개수를 축소시키지 않고 사례베이스의 크기를 최대로 유지하면서 분류를 수행하였다. 그 결과도 만족할만한 것이었지만, 문서의 속성 개수를 축소시키는 모듈이 포함된다면 좀더 분류시간을 단축시킬 수 있을 것이다. 둘째, CB_TFIDF 방법의 적중률이 100%가 아니므로 적중률을 높이기 위한 연구가 수행되어야 하겠다. 즉, CB_TFIDF의 두 개의 파라미터 n과 k를 다양하게 변화시키면서 그 성능을 분석하여 최적의 파라미터 집합을 찾아내거나, 분류결과에 대한 사용자의 응답(Feedback)을 받아서 그것을 사례기반 추론의 적응(Adaptation) 단계에 반영하는 방안을 고려할 수 있다.

본 연구에서 개발한 CB_TFIDF 방법은 문서분류 시스템뿐만 아니라, 문서 여과 시스템, 전자 문서관리 시스템, 개인화된 지능형 정보 검색 시스템 등에서 널리 활용될 수 있다. 특히 기존의 연구들이 주로 영어 문서들을 대상으로 한 반면, 본 연구는 한글 문서를 대상으로 하였으므로 그 의의가 있다고 할 수 있다.

〈참 고 문 헌〉

[1] 김시천, Memory-Based Reasoning을 이용한 HTML 문서분류 시스템의 설계 및 구축, 아주대학교 경영정보학과 석사학위 논문, 1999.

[2] 안수산, 신경식, "데이터마이닝 기법을 활용한 스팸메일의 분류 및 예측모형 구축에 관한 연구," 한국지능정보시스템학회 2000년 추계학술 대회 논문집, 2000, pp. 359-366.

[3] 이형일, 향상된 메모리기반 추론에 의한 지능형 문서여과 에이전트 구현, 명지대학교 컴퓨터 공학과 박사학위 논문, 1999.

[4] 한글공학 연구소, 한국어 분석 라이브러리 HAM 사용 설명서, 한성대학교, 1999.

[5] Aamodt, A. and E. Plaza, "Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches," *Artificial Intelligence Communications*, Vol. 7, No. 1, 1996, pp. 9-13.

[6] Baeza-Yates, R. and B. Ribeiro-Neto, *Modern Information Retrieval*, Addison Wesley, 1999.

[7] Cho, W.V., *Knowledge Discovery from Distributed and Textual Data*, Ph.D. Dissertation, Dept. of Computer Science, Hong Kong University of Science and Technology, 1999.

[8] Gudivada, V., V.V. Raghhavan, W.I. Grosky, and R. Kasanagottu, "Information Retrieval on the World Wide Web," *IEEE Internet Computing*, 1997.

[9] Joachims, T. A., "Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization," *Proc. 14th Int'l Conf. Machine Learning*, 1997, pp. 143-151.

[10] Kolodner, J., *Case-Based Reasoning*, Morgan Kaufman Pub. Inc., 1993.

[11] Lewis, D.D. and M. Ringuette, "Comparison of Two Learning Algorithms for Text Categorization," *Proc. 3rd Ann. Symp. Document Analysis and Information Retrieval*, 1994, pp. 81-93.

[12] Linoff, G. and M.J. A. Berry, *Mastering Data Mining*, Wiley, 2000.

[13] Mladenic, D., "Text-Learning and Related Intelligent Agents : A Survey," *IEEE Intelligent Systems*, 1999.

[14] Salton, G. and C. Buckley, "Term-weighting Approaches in Automatic Retrieval,"

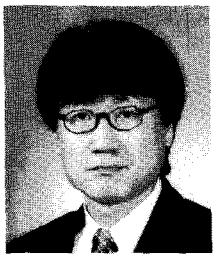
Information Processing and Management, Vol. 24, No. 5, 1988, pp. 513-523.

[15] Trybula, W.J., *Text Mining and Knowledge Discernment: An Exploratory Investigation*, Ph.D. Dissertation, The University of Texas at Austin, 1999.

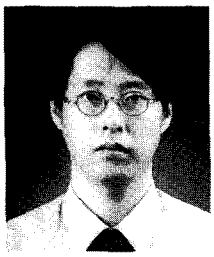
[16] Watson, I., *Applying Case-Based Reasoning: Techniques for Enterprise Systems*, Morgan Kaufman Pub. Inc., 1997.

[17] Weiner, E., J. O. Pedersen and A. S., Weigend, "A Neural Network Approach to Topic Spotting," *Proc. 4th Ann. Symp. Document Analysis and Information Retrieval*, 1995, pp. 197-208.

◆ 저자소개 ◆



이재식 (Jae Sik Lee)
현재 아주대학교 경영대학 경영학부의 교수로 재직중이다. 서울대학교 경영학과에서 경영학사(1977), 한국과학기술원 산업공학과에서 공학석사(1979), 그리고 University of Pennsylvania, Wharton School에서 경영정보시스템 전공으로 경영학 박사학위(1989)를 취득하였다. Management Science, Decision Support Systems, Expert Systems with Applications 등의 외국 학술지 및 다수의 국내 학술지에 논문을 게재하였다. 주요 관심분야는 Data Mining, AI Application to Business Problem Solving, Development of Intelligent Systems, Model Management Systems 등이다.



이종운 (Jong Woon Lee)
현재 대우정보시스템의 대우건설 SM팀에 재직하고 있으며, SAP 관리회계 시스템 운영 업무를 수행하고 있다. 아주대학교 경영학과를 졸업하고(1999), 동 대학원 경영정보학과에서 석사학위(2001)를 취득하였다. 주요 관심분야는 Data Mining 관련 Analytical 영역 CRM과 전략적 기업경영(SEM) 시스템이다.

◈ 이 논문은 2002년 3월 20일 접수하여 1차 수정을 거쳐 2002년 6월 10일 게재확정되었습니다.