■ 2002 정보과학 논문경진대회 수상작

베이지언 문서분류시스템을 위한 능동적 학습 기반의 학습문서집합 구성방법

(An Active Learning-based Method for Composing Training Document Set in Bayesian Text Classification Systems)

김 제 욱 † 김 한 준 †† 이 상 구 †††

(Je-uk Kim) (Han-joon Kim) (Sang-goo Lee)

요 약 기계학습 기법을 이용한 문서분류시스템의 정확도를 결정하는 요인 중 가장 중요한 것은 학습문서집합의 선택과 그것의 구성방법이다. 학습문서집합 선택의 문제란 임의의 문서공간에서 보다 정보량이 큰 적은 양의 문서집합을 골라서 학습문서로 채택하는 것을 말한다. 이렇게 선택한 학습문서집합을 재구성하여 보다 정확도가 높은 문서분류함수를 만드는 것이 학습문서집합 구성방법의 문제이다. 전자의 문제를 해결하는 대표적인 알고리즘이 능동적 학습(active learning) 알고리즘이고, 후자의 경우는 부스팅(boosting) 알고리즘이다.

본 논문에서는 이 두 알고리즘을 Naïve Bayes 문서분류 알고리즘에 적용해보고, 이때 생기는 여러 가지 특징들을 분석하여 새로운 학습문서집합 구성방법인 AdaBUS 알고리즘을 제안한다. 이 알고리즘은 능동적 학습 알고리즘의 아이디어를 이용하여 최종 문서분류함수를 만들기 위해 임시로 만든 여러 임시 문서분류함수(weak hypothesis)들 간의 변이(variance)를 높였다. 이를 통해 부스팅 알고리즘이 효과적으로 구동되기 위해 필요한 핵심 개념인 교란(perturbation)의 효과를 실현하여 문서분류의 정확도를 높일 수 있었다. Reuter-21578 문서집합을 이용한 경험적 실험을 통해, AdaBUS 알고리즘이 기존의 알고리즘에 비해 Naïve Bayes 알고리즘에 기반한 문서분류시스템의 정확도를 보다 크게 향상시킨다는 사실을 입증한다.

키워드 : 학습문서집합 구성방법, Naïve Bayes 문서분류 알고리즘, 부스팅 알고리즘, 불확실성 기반 샘플링 알고리즘, AdaBUS 알고리즘

Abstract There are two important problems in improving text classification systems based on machine learning approach. The first one, called "selection problem", is how to select a minimum number of informative documents from a given document collection. The second one, called "composition problem", is how to reorganize selected training documents so that they can fit an adopted learning method. The former problem is addressed in "active learning" algorithms, and the latter is discussed in "boosting" algorithms.

This paper proposes a new learning method, called AdaBUS, which proactively solves the above problems in the context of Naïve Bayes classification systems. The proposed method constructs more accurate classification hypothesis by increasing the variance in "weak" hypotheses that determine the final classification hypothesis. Consequently, the proposed algorithm yields perturbation effect makes the boosting algorithm work properly. Through the empirical experiment using the Reuters-21578 document collection, we show that the AdaBUS algorithm more significantly improves the Naïve Bayes-based classification system than other conventional learning methods

Key words : composing train document set, Naïve Bayes text classifier, boosting algorithm, uncertainty-based sampling algorithm, AdaBUS algorithm

† 비 회 원 : 대우정보시스템 기술연구소
jeuk@disc.co.kr
†† 비 회 원 : 서울대학교 공과대학 컴퓨터공학부
hjkim@europa.snu.ac.kr
††† 종신회원 : 서울대학교 공과대학 컴퓨터공학부 교수
sglee@europa.snu.ac.kr
논문접수 : 2002년 6월 19일
심사완료 : 2002년 9월 17일

1. 서 론

인터넷의 사용 증가 추세에 맞추어 전자 문서의 양은 폭발적으로 증가하고 있다. WWW에서 접할 수 있는 온라인 문서, 인터넷 뉴스 문서, 전자 메일 문서, 의료 정보 문서 그리고 디지털 도서관의 문서 등이 전자 문서에 속한다. 전자 문서가 양적으로 크게 늘어남에 따라 사람이 이러한 수많은 정보를 일일이 분류하는 것은 거의 불가능해졌다. 이에 따라 문서를 알맞게 분류하는 것을 도와주는 도구에 대한 요구가 점차 커지고 있다.

일반적으로 문서분류시스템(text classification system)이란, 텍스트 문서를 그것의 내용에 기반하여 미리 정해진 카테고리로 자동 분류하는 시스템을 말한다. 전자메일 분류시스템, 웹 문서 필터링 시스템 등이 문서분류시스템을 이용한 대표적인 응용 시스템이다. 사람은 문서분류시스템의 도움을 받아 수 많은 문서를 일일이 분류해야 하는 수고를 크게 덜 수 있다.

자동 문서분류를 위한 여러 가지 기법이 존재하는데, 규칙기반(rule-based) 기법 그리고 기계학습(machine learning) 기법 등이 대표적이다. 이 중에서도 기계학습 기법은 최근까지 매우 활발한 연구가 진행되고 있으며, 문서분류시스템의 문서분류 정확도(accuracy)를 크게 향상시키는데 기여하고 있다. 본 논문에서는 기계학습 기법을 이용한 문서분류의 문제를 다룬다. 이 기법에서는 학습 방법이 문서분류의 정확도를 크게 좌우하기 때문에 이에 대한 연구는 문서분류의 문제에 있어 상당히 중요하다.

문서분류시스템의 문서분류 정확도를 결정하는 요인을 두 가지 측면으로 요약할 수 있다. 첫 번째는 학습문서집합의 구성방법이다. 문서분류를 위해서는 임의의 문서공간 상에서 일정량의 문서집합을 택하여 학습문서집합을 구성한 후, 이를 이용하여 문서분류함수를 만든다. 따라서 어떤 문서를 학습문서로 선택할 것인가는 문서분류시스템에서 매우 중요하다. 학습문서집합의 구성방법과 관련하여 학습문서 선택의 문제 외에도 선택한 학습문서집합의 재구성 문제도 문서분류의 성능에 큰 영향을 미친다. 주어진 학습문서집합을 있는 그대로 이용하여 문서분류함수를 만들 수도 있으나, 이를 지능적으로 재구성하여 이용하면 보다 더 정확한 문서분류함수를 얻을 수 있을 것이다.

문서분류 시스템의 정확도를 결정하는 두 번째 요인으로 문서분류함수를 추정하는 알고리즘을 들 수 있다. 학습문서집합 구성방법에 의해 채택된 학습문서집합을 이용하여 최종적으로 문서분류함수를 만들어 내는 것이 이 알고리즘의 역할이다. 의사결정트리(decision tree), 알고리즘, k-최단 인접 기법(k-nearest neighbor), 알고리즘, SVM(support vector machine) 알고리즘, 신경망(neural networks)알고리즘, Naïve Bayes 알고리즘 등등이 이에 속한다.

본 논문에서는 학습문서집합 구성방법의 대표적인 방법인 능동적 학습 알고리즘과 부스팅 알고리즘을 다룬다. 전자는 학습문서를 선택하는데 초점을 맞춘 알고리즘이고, 후자는 학습문서집합을 재구성하여 더욱 정확도가 큰 문서분류함수를 만들어내기 위한 알고리즘이다. 이 두 가지 유형의 학습 알고리즘을 Naïve Bayes 문서분류 알고리즘에 적용해본다. 그리고 이 때 생기는 여러 가지 이슈들을 분석하여, 새로운 학습문서집합 구성방법인 AdaBUS(Adaptive Boosting with Uncertainty-based Sampling) 알고리즘을 제안한다. 이 알고리즘은 보다 더 적은 학습문서로 문서분류시스템의 정확도를 높이는 것을 목표로 한다.

본문은 다음과 같이 구성되어 있다. 2장에서는 문서분류시스템을 구성하는 요소와 문서분류시스템의 시스템 흐름도를 형식을 갖추어 정의해보고, 본 논문에서 채택한 문서분류함수 추정 알고리즘인 Naïve Bayes 알고리즘에 대해 살펴본다. 3장에서는 문서분류의 정확도를 높이는데 중요한 요소로 작용하는 학습문서집합 구성방법에 대해 논한다. 대표적인 학습문서집합 구성방법인 부스팅 알고리즘과 능동적 학습 알고리즘을 Naïve Bayes문서분류 알고리즘에 적용하는 방법을 살펴보고, 각 방법의 특성과 한계를 논한다. 4장에서는 3장에서 수행한 분석을 토대로 새로운 학습문서집합 구성 방법론인 AdaBUS 알고리즘을 소개한다. 5장에서는 Reuter-21578 문서 집합을 이용한 실험을 통해 본 논문에서 제시한 학습문서집합 구성방법이 기존의 방법보다 Naïve Bayes 문서분류 알고리즘의 문서분류 정확도를 더 크게 향상시킨다는 사실을 보인다. 마지막으로 6장에서는 본 논문의 결론을 내리고 향후 연구 방향에 대하여 논의한다.

2. 배경 이론

이 장에서는 기계학습 기반 문서분류시스템의 시스템 흐름도와 그 구성요소들을 살펴보고, 본 논문에서 문서분류함수 추정 알고리즘으로 선택한 Naïve Bayes 문서분류 알고리즘의 정의와 특징에 대하여 살펴본다.

2.1 문서분류시스템모델링의 구성

그림 1은 문서분류시스템의 전체 흐름도를 도식화한 것이다. 이를 자세히 살펴보자. 먼저 전체문서집합으로부터 일정한 개수의 문서를 선택한 후에 이에 대한 카

테고리를 전문가가 부여하여 학습문서집합을 구성한다. 이를 학습문서집합 구성방법을 통해 재구성하여 문서분류함수 추정 알고리즘에 제공한다. 이 때 최종적인 문서분류함수가 만들어진다. 문서분류함수를 이용하여 실질적인 문서분류를 수행한다. 문서분류시스템을 이루는 주요 구성요소는 다음과 같다.

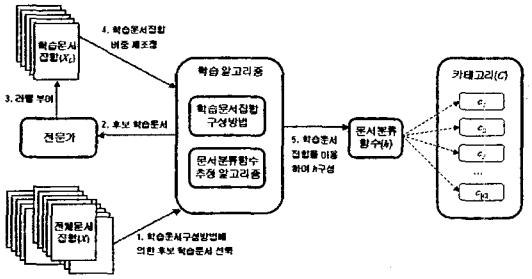

그림 1 문서분류시스템의 흐름도

정의 1. 카테고리 카테고리는 문서가 분류될 수 있는 정보를 가지고 있는 개념(concept) 또는 주제(topic)이다. 이러한 카테고리의 집합은 다음과 같이 표현한다.

$$C = \{c_1, c_2, \cdots, c_{|C|}\}$$

정의 2. 문서집합 본 논문에서 가정하는 문서분류시스템에서는, 시스템 내에 수 많은 문서가 존재한다고 가정한다 문서집합은 문서분류함수를 형성하는 기초가 되고 문서분류시스템의 정확도를 측정하는데 사용되는 시스템 내의 모든 문서들의 집합이다. 문서분류함수를 만드는데 사용되는 문서집합을 학습문서집합이라 하고, 정확도 측정에 사용되는 문서집합을 테스트 문서집합이라고 한다.

$X=\{x_1, x_2, \cdots, x_{|X|}\}$, *X는 문서 집합*

$X = X_L \cup X_T$ ($X_L \cap X_T = \emptyset$, X_L: *학습문서집합*, X_T: *테스트문서집합*)

문서집합에 속하는 각각의 문서는 반드시 하나의 카테고리에 속한다. 이를 형식적으로 나타내면 다음과 같다.

X에 속하는 모든 문서 x에 대하여,

$c(x) \in C$ (*단, $c(x)$는 x가 속하는 카테고리*)

정의 3. 전문가 문서분류시스템에서 전문가(oracle)는 문서의 실제 카테고리를 판단할 수 있는 사람이라고 정의된다. 즉, 전문가는 문서 x의 실제 카테고리인 $c(x)$를 판단한다. 따라서 전문가는 문서분류시스템의 초기 학습을 담당한다. 이를 형식적으로 표현하면 다음과 같다.

$$o: X \rightarrow C$$

정의 4. 문서분류함수 문서분류함수(hypothesis)는 입력으로 주어진 문서의 카테고리를 추정하는 함수이다. 따라서 다음과 같은 정의가 가능하다.

$$h: X \rightarrow C$$

문서분류함수 h는 문서의 카테고리를 추정한다. 따라서 임의의 문서 x ($\in$X)에 대하여 만약 $h(x)=c(x)$ 이면 h는 x의 카테고리를 정확히 분류한 것이고, 그렇지 않으면 틀리게 분류한 것이다.

정의 5. 문서분류함수 추정 알고리즘 문서분류함수 추정 알고리즘이란, 주어진 학습문서집합을 이용하여 문서분류함수를 만들어내는 알고리즘을 말한다. 예를 들어 의사결정트리, k-최단 인접 기법, 신경망 그리고 Naïve Bayes 문서분류 알고리즘 등이 이에 해당된다.

정의 6. 학습문서집합 구성방법 학습문서집합 구성방법이란, 학습문서가 될 문서들을 전체문서집합으로부터 선택하고, 선택한 학습문서집합을 재구성하여 문서분류함수 추정 알고리즘에 제공하는 일련의 알고리즘을 말한다. 간단히 전체문서집합으로부터 임의로 학습문서를 선택하여 학습문서집합을 구성한 후 이를 문서분류함수 추정 알고리즘에 그대로 제공하는 것도 학습문서집합 구성방법의 예이다.

정의 7. 학습 알고리즘 학습 알고리즘은 학습문서집합 구성방법과 문서분류함수 추정 알고리즘을 합친 것을 말한다. 전체문서집합으로부터 학습문서집합을 골라내고 이를 이용하여 최종 문서분류함수를 만들어내는 것이 학습 알고리즘의 역할이다.

2.2 Naïve Bayes 문서분류 시스템알고리즘

Naïve Bayes 문서분류 알고리즘은 분류하려는 문서를 입력으로 받아 그것이 각 카테고리에 할당될 확률을 계산하는 방법으로 문서를 분류한다. 문서가 특정 카테고리에 속하는 확률을 계산하기 위하여 식 (1)의 베이즈 이론(Bayes' theorem)을 이용한다[1].

$$P(c \mid x) = \frac{P(c)P(x \mid c)}{P(x)} \quad (1)$$

$$P(x) = \sum_{c \in C} P(c)P(x \mid c) \quad (2)$$

여기에서 x는 임의의 문서를 의미하고 c는 임의의 카테고리를 의미한다. 식 (1)의 $P(x)$는 전확률 공식(total probability formula)에 의해 식 (2)와 같이 정의된다. 그런데 $P(x)$는 모든 카테고리에 대하여 같은 값을 가지므로 확률을 계산하는데 고려하지 않아도 된다. 따라서 식 (1)의 분모에 위치한 $P(c)$와 $P(x|c)$만 추정하면 문서 x가 카테고리 c에 할당될 확률을 계산할 수 있다. $P(c)$는 모든 카테고리 중 카테고리 c가 뽑힐 확률이다. 따라서 이는 모든 학습 문서들의 수인 $|X_L|$와 카테고리 c에 속하는 학습 문서들의 수인 $|X_{L,c}|$의 비율로 추정할 수 있다. 따라서 다음과 같은 식이 성립한다.

$$P(c) = \frac{|X_{L,c}|}{|X_L|} \quad (3)$$

$P(c|x)$를 계산하기 위해서는 우리는 마지막으로 $P(x|c)$를 계산해야 한다. 문서 x는 단어들의 벡터인 $<w_1, w_2, \cdots, w_{|x|}>$로 나타낼 수 있다. 따라서 $P(x|c)$는 다시 $P(<w_1, w_2, \cdots, w_{|x|}>|c)$로 나타낼 수 있다. Naïve Bayes 문서분류 알고리즘은 $P(<w_1, w_2, \cdots, w_{|x|}>|c)$의 계산을 좀 더 쉽게 하기 위해, 문서 내에 존재하는 모든 단어들인 $w_1, w_2, \cdots, w_{|w|}$가 서로 독립(independent)이고, 문서 내의 단어 위치와 그 단어의 출현확률사이에도 독립성이 존재한다고 가정한다. 이 가정에 따르면 $P(x|c)$는 다음과 같은 식으로 표현된다.

$$P(x \mid c) = \prod_{k=1}^{|x|} P(W_k \mid c) \quad (4)$$

n_c를 c카테고리에 출현하는 모든 단어들의 빈도수의 합이라 하고, $n_{c,w}$를 c카테고리에 출현하는 w단어의 빈도수라 할 때, $P(w|c)$의 추정치는 $\frac{n_{c,w}}{n_c}$이라 할 수 있다. 그러나 이 추정치를 식 (4)에 그대로 적용하면, 이 전체 식의 값을 0으로 만들 확률이 높다. 왜냐하면, 분류하려는 문서 내에 존재하는 단어가 확률을 계산하려는 카테고리 내에 존재하지 않을 수도 있기 때문이다. 이러한 문제를 해결하기 위해서 일반적으로 식 (5)와 같이 *m-estimate* 개념을 응용한 기법을 이용한다[1,2]. 여기에서 $|vocabulary|$는 모든 학습문서 내에 포함되어 있는 서로 다른 단어의 개수이다.

$$P(w \mid c) = \frac{n_{c,w}+1}{n_c+|vocabulary|} \quad (5)$$

Naïve Bayes 문서분류 알고리즘을 이용한 문서분류 시스템은 일반적으로 다른 알고리즘들(의사결정트리, k-최단 인접 기법, 신경망 알고리즘 등)에 비해 문서분류의 정확도가 상대적으로 높다고 알려져 있다[3]. 또한 Naïve Bayes 문서분류 알고리즘은 이 알고리즘에 따르면 문서분류함수의 구축이 간단하며, 고 문서분류의 속도가상대적으로 빠르기 때문에 문서분류시스템을 구축에 매우 빈번히 이용된다. 따라서 이 알고리즘을 토대로 한을 택한 문서분류시스템의 문서 정확도를 높이는 작업은 큰 의미가 있다고 할 수 있다.

본 논문은 Naïve Bayes 문서분류 알고리즘에 제공하는 학습문서집합을 지능적으로 구성하여 Naïve Bayes 문서분류 알고리즘이 추정하는 문서분류함수의 문서분류 정확도를 보다 향상시키는 것을 목표로 한다.

3. 학습문서집합 구성방법론

3.1 학습문서집합 구성방법의 개념

2.1절에서 학습문서집합 구성방법의 개략적인 정의를 다루었다. 여기서는 이를 좀 더 구체적으로 살펴본다. 학습문서집합 구성방법이란 학습문서집합과 문서분류함수 추정 알고리즘을 이용하여 문서분류함수를 만들어내는 일련의 알고리즘을 의미한다. 이를 그림 2에 도식화하여 나타내었다. 이를 자세히 살펴보자. 학습문서집합 구성방법은 세 가지 작업으로 분류된다. 첫째는 전체 문서집합으로부터 학습문서집합을 골라내는 작업이다. 둘째는 그리고 이렇게 주어진 학습문서집합을 재구성하여 문서분류 추정 알고리즘에 제공하는 것이다. 셋째는 이러한 과정을 여러 번 거쳐 여러 개의 문서분류함수들을 만든 후에 이들을 이용하여 하나의 새로운 문서분류함수를 만드는 것이다.

학습문서집합 구성방법에 있어서 쟁점이 되는 문제는 다음 두 가지이다. 첫째는 학습문서집합의 선택의 문제로서, 전체 문서집합으로부터 정보량이 큰 문서를 골라 이를 학습문서로 채택하는 문제가 바로 그것이다[4][5]. 능동적 학습 알고리즘이 이러한 문제를 다루는 대표적인 알고리즘이다. 두 번째 문제는 주어진 학습문서집합을 재구성하여 보다 정확도가 높은 문서분류함수를 만들어내는 것이다. 대표적으로 부스팅 알고리즘에서는 주어진 학습문서집합에 속한 각 학습문서의 비중(weight)을 달리하여 여러 개의 학습문서집합을 만든다. 그 후 이들 각 학습문서집합을 이용하여 여러 개의 문서분류함수를 만든 후에 투표(voting)방법을 이용하여 이들을 합친 후에 하나의 문서분류함수를 만든다. 부스팅 알고리즘에 따르면, 이렇게 만든 최종 문서분류함수의 문서분류 정확도는 그 전에 만든 여러 개의 임시 문서분류함수 각각의 정확도보다 클 것이라고 기대한다[6].

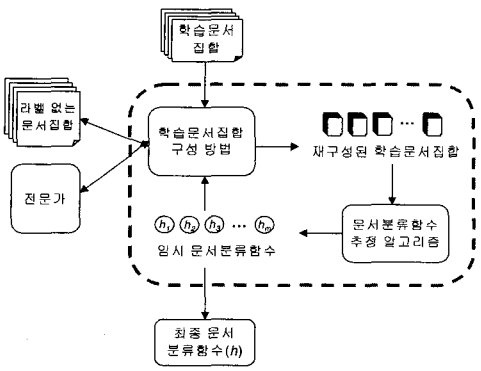

그림 2 학습문서집합 구성방법의 개념

이제 다음 절에서는 학습문서집합 구성방법의 대표적인 두 알고리즘인 능동적 학습 알고리즘과 부스팅 알고리즘에 대하여 살펴본다.

3.2 능동적 학습 알고리즘

3.2.1. 기본 개념

능동적 학습 알고리즘은 전체문서집합으로부터 정보량이 큰 문서를 선택하여 이를 학습문서집합에 추가한다. 이를 통해 문서분류함수의 정확도를 높이는 것이 이 알고리즘의 목적이다.

능동적 학습 알고리즘에서 정보량이 큰 문서를 판단하는 기준은 여러 가지이다.도 다양한데, 그 중 대표적인 것이 불확실성 개념을 이용하는 것이다[4,5,7,8]. 능동적 학습 알고리즘 중에서 이 개념을 이용하는 알고리즘을 특히 불확실성 기반 샘플링 알고리즘이라고 한다. 이에 따르면 정보량이 큰 문서는 현재의 문서분류함수가 분류하기 어려운(또는 애매한) 문서이다. 문서분류함수가 분류하기 어려운 문서란, 문서분류함수가 입력으로 주어진 문서의 카테고리를 판단할 때 확신이 작작은 문서를 의미한다. 이러한 문서는 카테고리와 카테고리를 나누는 경계(또는 경계 부근)에 위치하고 있기 때문에 확실히 어떠한 카테고리에 할당되어야 하는지 판단하기 어려운 성질을 지닌다.

능동적 학습 알고리즘의 개념을 그림 3을 통해 살펴보자. 그림에서 큰 동그라미와 삼각형은 각각 c_1, c_2 카테고리에 속하는 학습문서이다. 그리고 작은 동그라미로 표시된 것은 아직 라벨이 결정되지 않은 문서를 의미한다. 점선은 실제 카테고리를 나누는 경계를 의미하고 실선은 현재 문서분류함수가 카테고리를 분류하는 경계를 의미한다. 이러한 상황을 가정할 때, 정보량이 크기 때문에 학습문서로 채택되면 현재 문서분류함수의 정확도를 좀더 커지게 하는 문서는 어떤 문서일까? 가장 분류하기 어려운 문서들인 x_1, x_2가 이에 해당될 것이다. 이들을 학습문서로 채택하여 정확한 카테고리를 할당한다면, 그림과 같이 문서분류함수가 문서를 분류하는 기준으로 삼고 있는 카테고리간의 경계가 좀 더 실제 경계와 가까워진다.

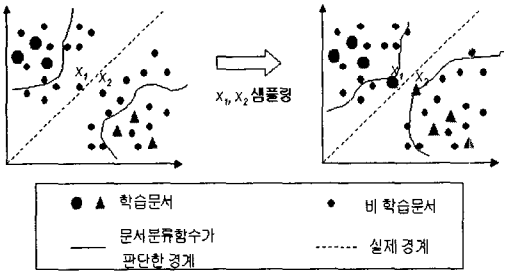

그림 3 불확실성 기반 샘플링 알고리즘의 개념

3.2.2 불확실성 기반 샘플링 알고리즘

본 절에서는 3.2.1절에서 개략적으로 설명한 불확실성 기반 샘플링 알고리즘을 자세히 살펴본다. 그림 4는 이 알고리즘의 의사코드를 보여준다.

```
1. 입력: 초기학습문서집합 D_T={<x_1, c(x_1)>, ..., <x_m, c(x_m)>}
         라벨이 없는 문서집합 X_U
2. D_T로부터 문서분류추정함수를 만든다. h_t ← L(D_T)
3. For t=1 ... m:
   3-a) h_t를 이용하여 X_U로부터 불확실성이 가장 큰 문서를 선택한다.
        x ← U(X_U)
   3-b) 전문가가 문서 x의 라벨을 부여한다.  c_x ← c(x)
   3-c) x를 학습문서집합에 추가한다.
        D_T ← D_T ∪ {<x, c_x>}
   3-d) 새로운 문서분류함수를 만든다 h_{t+1} ← L(D_T)
4. 마지막 문서분류함수 h_{m+1}를 최종 문서분류함수로 한다
```

그림 4 불확실성 기반 샘플링 알고리즘의 의사코드

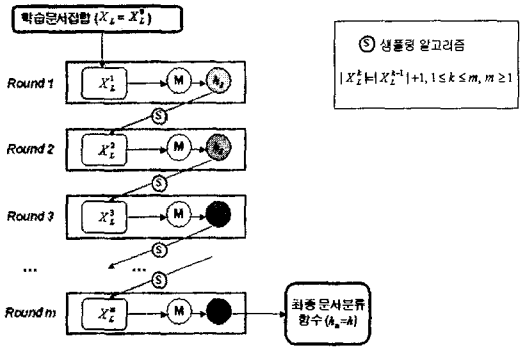

그림 5 불확실성 기반 샘플링 알고리즘의 도식화

그림 4에 제시된 의사코드를 자세히 살펴보자. D_T는 초기에 주어진 학습문서집합이고, X_U는 라벨이 없는 문서들의 집합이다. $U(X_U)$는 문서 집합 X_U로부터 정보량이 가장 큰 문서, 즉 가장 애매한 문서를 선택하는 함수이다. 이 함수는 불확실성이 가장 큰 문서를 찾아내는 역할을 한다. L은 학습문서집합으로부터 문서분류함수를 만들어내는 문서분류함수 추정 알고리즘이다. 예를 들어 Naïve Bayes 문서분류 알고리즘이 이에 해당한다. h는 D_T로부터 만들어진 문서분류함수이다.

불확실성 기반 샘플링 알고리즘은 우선 초기학습문서집합을 이용하여 초기 문서분류함수를 만드는 것으로 시작한다(line 2행). 그 후 종결조건이 만족될 때까지 점진적으로 학습을 진행한다(3행). 라벨이 없는 문서의 집합으로부터 정보량이 큰 문서를 선택한 다음(3-a행), 이 문서에 라벨을 부여한다(line 3-b행). 이 문서를 D_T에 추가하고(3-c행), 현재의 문서분류함수를 바꾼 D_T를 이

용하여 다시 만든다(3-d행). 종결 조건이 만족되면 최종 문서분류함수가 만들어지면서 알고리즘은 종료된다(4행). 여기서 종결조건은 전문가가 문서에 라벨을 부여하는 것을 중단하는 시점을 의미한다. 그림 5에서는 그림 4에 제시된 의사코드를 알기 쉽게 도식화한 것이다.

3.2.3 Naïve Bayes 문서분류 알고리즘으로의 적용

2.2절에서 소개했듯이, Naïve Bayes 문서분류 알고리즘은 분류하려는 문서가 각 카테고리에 속할 확률을 계산하는 방식으로 문서를 분류한다. Naïve Bayes 문서분류 알고리즘의 이러한 성질을 이용하여 불확실성을 다양한 방식으로 정의할 수 있다[4].

불확실성이란 특정 문서가 문서분류함수에 의해 분류되는 경우, 얼마나 불명확하게 분류되는가를 측정한 수치를 말한다[4,5,7]. 따라서 불확실성이 클수록 문서분류함수가 해당 문서를 카테고리로 분류하는 확신이 작다. 위에서 살펴보았듯이 불확실성 기반 샘플링 알고리즘은 라벨이 없는 문서집합 내의 모든 문서의 불확실성을 측정한 후에 가장 불확실성이 큰 문서를 골라서 이를 학습문서로 채택한다. 불확실성은 문서의 카테고리를 예측하고, 이 예측에 대한 확신을 수치로 나타낼 수 있는 문서분류 알고리즘에서는 모두 정의가 가능하다. 여기서는 Naïve Bayes 문서분류 알고리즘에서 불확실성을 측정하는 두 가지 측정치를 소개한다.

첫 번째로 신뢰도(confidence) 측정치를 살펴보자. 문서 x가 카테고리 c_i로 할당되는 경우의 신뢰도는 다음과 같이 정의된다[4,8].

$$U_{confidence}(x)=\frac{P(c_i\mid x)-P(c_j\mid x)}{P(c_i\mid x)} \tag{6}$$

이 식에서 는 문서 x가 카테고리 c_i에 속할 확률을 의미한다. 이는 다음과 같은 성질을 갖는다.

$$P(c_1\mid x)+P(c_2\mid x)+\cdots+P(c_{|C|}\mid x)=1,\quad \text{모든 } c\in C\text{에 대하여, } 0\le P(c\mid x)\le 1 \tag{7}$$

식 (6)에서 c_i는 문서 x와 가장 가까운 카테고리이다. 즉, c_i는 $P(c|x)$ 값을 가장 크게 하는 카테고리이다. 또한 c_j는 문서 x와 두 번째로 가까운 카테고리이다. $P(c_i|x)-P(c_j|x)$ 값이 클수록 현재 문서분류함수가 분류 결과에 대한 확신을 크게 갖고 있다는 의미이므로, 식 (6)의 신뢰도 값은 커진다. 이와 같이 신뢰도와 불확실성은 반비례의 관계에 있다. 신뢰도가 크다는 것은 문서분류함수가 문서를 정확하게 분류할 가능성이 크다는 것을 의미하기 때문이다. 그러므로 신뢰도의 역수를 이용하여 불확실성을 측정할 수 있다.

두 번째로 평균절대편차(MAD, Mean Absolute Deviation)를 이용한 불확실성 측정치를 알아보자[4]. 이 측정치에서는 앞에서 정의한 $P(c|x)$의 값들이 그 값들의 평균(μ)과 얼마나 떨어져 있는지를 이용하여 불확실성을 측정한다. 이는 다음과 같이 정의된다.

$$U_{MAD}(x)=\frac{1}{|C|}\sum_{i=1}^{|C|}(P(c_i|x)-\mu) \tag{8}$$

$$\mu=\frac{1}{|C|}\sum_{i=1}^{|C|}P(c_i|x) \tag{9}$$

$U_{MAD}(x)$는 문서 x가 각 카테고리에 속할 확률 또는 소속값들이 그 값들의 평균으로부터 떨어진 평균거리를 의미하므로 $U_{MAD}(x)$가 작을수록 불확실성이 크다고 할 수 있고, 클수록 불확실성이 작다고 할 수 있다.

[4]에서는 불확실성 기반 샘플링 알고리즘을 Naïve Bayes 문서분류 알고리즘에 적용하여 문서분류함수의 정확도를 크게 향상시켰다. [4]에서는 불확실성의 측정치로서 앞에서 설명한 신뢰도 측정치와 평균절대편차 측정치를 모두 이용하여 실험을 하였는데, 평균절대편차 측정치를 이용했을 경우의 문서분류 정확도가 신뢰도 측정치를 이용한 경우보다 더 컸다.

3.3 부스팅 알고리즘

3.3.1 기본 개념

부스팅 알고리즘의 주요 아이디어는 주어진 학습문서집합을 적절히 조작하여 서로 다른 여러 개의 문서분류함수들을 만들고, 이들을 합쳐서 하나의 성능이 좋은 문서분류함수를 만드는 것이다[6,9]. 이 알고리즘은 주어진 학습문서집합 내의 학습문서의 비중을 특정 규칙에 의해 변화시킨다. 이렇게 재구성한 학습문서집합을 이용하여 임시 문서분류함수를 만든다. 이런 과정을 여러 번 거쳐서 만든 임시 문서분류함수들을 합쳐 하나의 문서분류함수를 만들어낸다. 합치는 방법은 일반적으로 투표(majority voting)방법을 이용한다. 이 방법에 따르면 분류하려는 문서를 여러 개의 문서분류함수를 이용하여 분류를 한 후에, 각각의 분류결과를 합산하여 가장 많이 할당된 카테고리로 해당 문서를 분류한다.

부스팅 알고리즘의 핵심 개념은 교란(perturbation)과 합성(combining)이다[6,10]. 교란은 여러 개의 임

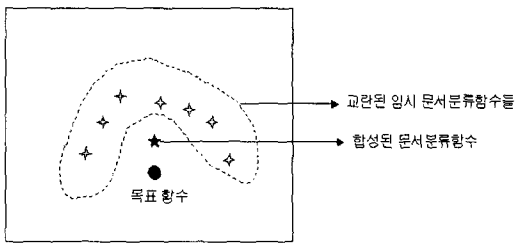

그림 6 부스팅 알고리즘의 개념

시 문서분류함수를 만드는 과정인데, 이때 이들 임시 문서분류함수들 각각이 서로 비슷하다면 교란을 제대로 실현하지 못한 것이다. 서로 이질적인 문서분류함수들을 만드는 것이 교란의 목적이기 때문이다. 합성은 이들 임시 문서분류함수들을 합쳐서 하나의 최종 문서분류함수를 만드는 과정이다. 교란이 적절히 이루어져서 서로 이질적인 함수들이 만들어져야만 합성의 결과로 만들어진 최종 문서분류함수의 문서분류 정확도의 향상을 기대할 수 있다. 그림 6은 이러한 개념을 도식화한 것이다. 그림에 소개된 공간은 가설 공간이다. 이 공간상의 플러스기호(+)는 교란에 의해 만들어진 여러 임시 문서분류함수를 의미한다. 이들 간의 거리가 멀수록 함수들이 서로 이질적임을 의미한다. 공간상의 별모양 기호는 이들 함수들을 합쳐서 만든 결과인 최종 문서분류함수를 의미한다. 그리고 원으로 표시된 기호는 목표 함수(target hypothesis)이다. 목표 함수란, 모든 문서에 대하여 올바른 카테고리를 추정하는 이상적인 문서분류함수를 의미한다. 따라서 가설 공간상의 문서분류함수가 목표 함수에 가까울수록 그것의 분류 정확도는 높다. 다음 절에서는 대표적인 부스팅 알고리즘인 AdaBoost 알고리즘을 살펴본다.

3.3.2 AdaBoost 알고리즘

AdaBoost알고리즘은 [11]에서 처음 제안되었다. 그 후 이 알고리즘은 수 많은 연구에서 수학적으로 분석되었고, 이 알고리즘을 적용하면 문서분류시스템의 정확도가 향상된다는 사실이 입증되었다[6,12,13]. 이 알고리즘의 기본 아이디어는 여러 개의 '정확도가 높지 않은 문서분류함수'(weak hypothesis)를 순차적으로 만들고, 이들을 합쳐서(combine) 하나의 성능이 좋은 최종 문서분류함수를 만드는 것이다. 학습문서의 비중을 조절할 때, 문서분류의 결과가 틀린 문서의 비중을 높이는 방식으로 알고리즘을 진행시켜 나가는 것이 AdaBoost 알고리즘의 중요한 특징이다.

이제 AdaBoost 알고리즘을 그림 7에 제시된 의사코드를 통해 자세히 살펴보자. 우선 알고리즘은 일정한 수의 학습문서집합 $D_T=\{<x_1, c(x_1)>, <x_2, c(x_2)>, \cdots, <x_m, c(x_m)>\}$를 입력으로 받는다(line 1행). 이때 각 학습문서는 미리 정해진 카테고리들 중 하나에 속한다. 그 후 각 학습문서의 비중을 모두 1/m으로 초기화시킨다(2행). 알고리즘이 진행될수록 각 학습문서의 비중은 재조정(re-weight) 전략에 따라 변한다. 이제그 후 알고리즘은 임시 문서분류함수를 t개 만큼 만들기 위해 t번의 라운드를 거쳐 알고리즘을 진행시킨다(3행).

3-a행에서는 이전 라운드에서 만든 학습문서들과 그들 각각이 가진 비중을 이용하여 현재 라운드의 임시 문서분류함수를 만든다. 이 분류함수를 이용하여 각 학습문서들 각각을 분류하여, 해당 라운드 분류함수(h_t)의 에러()를 계산한다(3-b행). 3-c행에서는 3-b행에서 구한 에러를 이용하여 해당 문서분류함수의 신뢰도인 α_t를 계산한다. α_t은 ε_t가 작을수록 커지는 값으로서, 나중에 각 라운드의 임시 문서분류함수를 합쳐 최종 문서분류함수를 만들 때 투표 비중(voting weight)으로도 이용된다. 3-d행에서는 앞에서 계산한 값들을 이용하여 각 학습문서의 비중을 변화시킨다. 문서분류의 결과가 옳은 경우($h_t(d_i)=c(d_i)$)에는 해당 문서의 비중에 $e^{-\alpha_t}$를 곱해줌으로써 비중을 낮추어주고, 반대의 경우에는 해당 문서의 비중에 e^{α_t}를 곱해서 그 만큼의 비중을 높여준다. 결과적으로 다음 라운드에 만들어지는 임시문서분류함수는 이전 분류함수에 적용하여 틀리게 분류된 학습문서의 비중을 좀더 높인 학습문서집합을 이용하여 만들어지게 된다. 마지막으로 4행에서는 이전에 만들었던 t개의 임시 문서분류함수를 이용하여 최종 문서분류함수를 만든다. 그림 8은 AdaBoost 알고리즘의 개념을 도식화하여 보여준다.

1. 입력: 초기 학습문서집합 $D_T=\{ <x_1, c(x_1)>, <x_2, c(x_2)>, \cdots, <x_m, c(x_m)> \}$
2. 모든 $x(D_T$에 대하여, $W_1(x_i) = 1/m$
3. For $t=1, \cdots, T$:
   - 3-a) D_T와 그것들의 비중 분포인 W_t를 이용하여 h_t를 만든다.
   - 3-b) h_t의 에러(error)를 다음과 같이 구한다.

$$\varepsilon_t = \sum_{x_i \in D_T} W_t(x_i)\delta(x_i), \quad \delta(x_i) = \begin{cases} 0, & if\ h_t(x_i) = c(x_i) \\ 1, & if\ h_t(x_i) \neq c(x_i) \end{cases}$$

   - 3-c) h_t의 신뢰도인 $\alpha_t = \frac{1}{2}\ln(\frac{1-\varepsilon_t}{\varepsilon_t})$를 구한다.
   - 3-d) 비중 재조정:

$$W_{t+1}(i) = \frac{W_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t}, & if\ h_t(x_i) = c(x_i) \\ e^{\alpha_t}, & if\ h_t(x_i) \neq c(x_i) \end{cases}$$

   여기서 Z_t 는 가 분포가 되도록 하기위한 정규화인수이다
4. 투표 방법을 이용하여 최종 문서분류함수를 구한다:

$$h(x) = voting(<h_1, \alpha_1, x>, \cdots, <h_t, \alpha_t, x>)$$

그림 7 AdaBoost 알고리즘의 의사코드

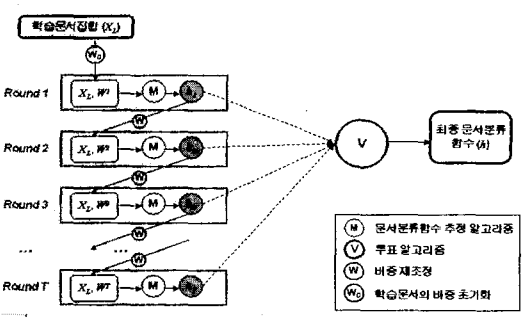

그림 8 AdaBoost 알고리즘의 도식화

최종 분류함수를 만드는 투표방법을 식 (10)에 나타내었다. 문서 x가 임의의 카테고리 c에 속한다고 임의의 임시 문서분류함수인 $h_t(h_t(\{h_1, h_2, \cdots, h_T\}))$가 판단했을 경우 h_t는 이것의 신뢰도인 만큼의 권한을 행사할 수 있다. 이 개념을 이용하여 모든 카테고리에 대하여 각 임시 문서분류함수가 투표할 결과를 합산하고, 가장 큰 값을 갖는 카테고리를 문서 x의 문서분류 결과로 추정한다.

$$h_{final}(x)=\arg\max_{c_i\in C}\sum_{t=1}^{T}\phi_t(c_i)\alpha_t,$$
$$where\ \phi_t(c_i)=\begin{cases}0, if\ h_t(x)\neq c_i\\ 1, if\ h_t(x)=c_i\end{cases} \quad (10)$$

3.3.3. Naïve Bayes 문서분류 알고리즘으로의 적용

3.3.1절에서 보았듯이 부스팅 알고리즘의 핵심은 교란과 합성이다. 특히 교란이 적절히 실현되지 않으면, 부스팅 알고리즘의 효과는 기대할 수가 없다. 교란이 적절히 실현된다는 말의 의미는 만들어낸 여러 개의 임시 문서분류함수들 간의 이질성이 크다라는 것을 말한다. [6][14]에서는 AdaBoost 알고리즘을 의사결정트리 문서분류 알고리즘에 적용하였는데, 의사결정트리 알고리즘의 성격 상 교란의 효과를 크게 얻을 수 있었기 때문에, AdaBoost 알고리즘을 적용한 결과 의사결정트리 알고리즘의 문서분류 정확도가 크게 향상되었다. 의사결정트리 알고리즘은 일반적으로 변이(variance)가 큰 알고리즘이라고 알려져 있는데, 그 이유는 학습문서집합의 조그마한 차이에도 의사결정트리 알고리즘이 만들어 내는 문서분류함수가 크게 영향을 받기 때문이다. 이러한 성질 때문에 학습문서집합의 비중을 달리해서 만들어낸 임시 문서분류함수간의 이질성을 크게 할 수 있었다.

반면 Naïve Bayes 문서분류 알고리즘은 상당히 안정적인 알고리즘으로 알려져 있다[15,16]. 여기서 안정적이라는 것의 의미는 Naïve Bayes 알고리즘이 만들어 내는 문서분류함수가 학습문서집합에 크게 영향을 받지 않는다는 것을 의미한다. 따라서 학습문서집합의 비중을 바꾼다고 하더라도 이에 기초하여 Naïve Bayes 알고리즘이 만들어 내는 문서분류함수는 비중을 바꾸기 이전의 문서분류함수와 크게 다르지 않다. 이러한 성질 때문에 Naïve Bayes 문서분류 알고리즘에 AdaBoost 알고리즘을 적용하면, 교란의 효과를 얻기가 상당히 어렵다. 따라서 AdaBoost 알고리즘을 적용하여 만들어 낸 최종 문서분류함수의 정확도를 향상시키는 것은 거의 불가능하다고 알려져 있다[14,15,16]. 결국 교란의 효과를 얻기 위해서 Naïve Bayes 문서분류 알고리즘의 안정성을 극복하기 위한 방법이 마련되지 않는 한 부스팅 알고리즘을 적용한 Naïve Bayes 문서분류 알고리즘의 성능향상을 기대할 수 없다.

4. 새로운 학습문서집합 구성방법: AdaBUS 알고리즘

4.1 기본 개념

3.2, 3.3절에서 소개한 능동적 학습 알고리즘과 부스팅 알고리즘은 모두 문서분류함수의 정확도를 높이기 위한 알고리즘이다. 이번 절에서는 이 두 알고리즘의 장점을 모두 수용한 알고리즘을 제시한다. AdaBUS(Adaptive Boosting with Uncertainty-based Sampling) 알고리즘이 그것이다. 2.2절에서 언급했듯이 Naïve Bayes 문서분류 알고리즘은 안정적인 알고리즘이기 때문에 부스팅 알고리즘을 Naïve Bayes 문서분류 알고리즘에 적용하여 문서분류의 정확도를 높이는 것을 기대하기는 어렵다. 부스팅 알고리즘을 적용하여 문서분류의 정확도를 높이기 위해서는 3.3.1절에서도 살펴보았듯이 무엇보다도 교란과 합성이 중요하다. 하지만 Naïve Bayes 문서분류 알고리즘의 안정성으로 인해 교란을 실현하기가 어렵다. AdaBUS 알고리즘은 불확실성 기반 샘플링 알고리즘의 개념을 도입하여 교란을 효과적으로 수행한다.

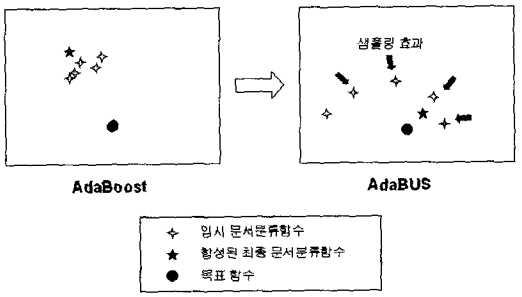

그림 9 AdaBUS 알고리즘의 기대효과

그림 9에 AdaBUS 알고리즘의 기대효과를 개념적으로 도식화 하였다. 그림 내에 있는 기호의 의미는 그림 6의 경우와 같다. 그림에서 보듯이 AdaBUS 알고리즘은 임시 문서분류함수들 간의 교란을 적절히 실현하여 최종 문서분류함수의 정확도를 높일 것으로 기대된다.

AdaBUS 알고리즘의 핵심 아이디어는 각 임시 문서분류함수를 같은 학습문서집합에 기반하여 만드는 것이 아니라, 불확실성 기반 샘플링 알고리즘을 이용하여 학습문서집합을 늘려가면서 임시 문서분류함수를 만드는

데에 있다. 불확실성 기반 샘플링 알고리즘을 통해 정보량이 큰 문서를 학습문서집합에 추가해가면서 임시 문서분류함수를 만든다면, 이들은 샘플링 효과에 의해서 좀 더 목표 함수에 가까워질 것이다. 따라서 그림과 같이 각 임시 문서분류함수들의 평균 문서분류 정확도가도 향상되고, 각 임시 문서분류함수들이 서로 이질적이 되어 결국 교란의 효과를 자연스럽게 얻을 수 있을 것이다. 즉, 부스팅 알고리즘에서 기대하는 효과를 얻을 수 있는 것이다.

그림 10에 AdaBUS 알고리즘의 개념을 도식화하여 나타내었다. 이 알고리즘은 부스팅 알고리즘에서 채택한 교란과 합성의 개념과 불확실성 기반 샘플링 알고리즘의 핵심 개념인 샘플링을 도입하여 최종문서분류함수를 만든다. 그림에서 볼 수 있듯이 이 알고리즘에서는 샘플링을 진행하면서 임시 문서분류함수들을 만들기 때문에 이들간의 이질성을 높이는 효과를 얻을 수 있다.

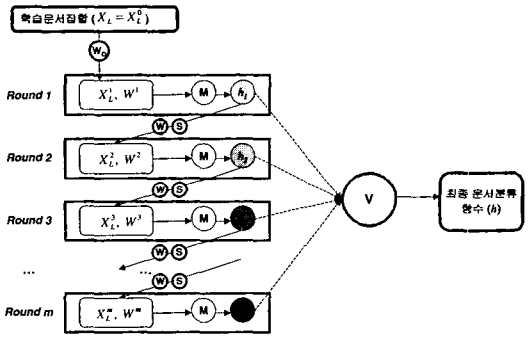

그림 10 AdaBUS 알고리즘의 도식화

4.2 임시 문서분류함수들 간의 변이 측정치

3.3절과 4.1절에서 살펴보았듯이 AdaBoost 알고리즘과 AdaBUS 알고리즘은 여러 개의 임시 문서분류함수들을 만들고 이들을 투표방법에 의해 합쳐서 하나의 최종 문서분류함수를 만든다. 이 때 임시 문서분류함수들 간의 변이가 커야 교란과 합성의 효과가 얻을 수 있다고 하였다. 이 절에서는 임시 문서분류함수들 간의 변이를 측정하는 식에 대하여 알아본다. 5장의 실험부분에서도 살펴보겠지만, AdaBUS 알고리즘에서는 임시 문서분류함수들 간의 변이가 AdaBoost 알고리즘의 경우보다 크기 때문에 교란을 보다 더 잘 실현할 수 있다.

임시 문서분류함수들 간의 변이는 각 테스트문서에 대하여 임시 문서분류함수들 각각이 얼마나 다르게 문서를 분류하느냐를 측정함으로써 추정할 수 있다[17]. 모든 테스트문서에 대하여 각 임시 문서분류함수들이 똑같은 카테고리로 분류를 한다면, 임시 문서분류함수들 간의 변이는 0이다. 변이를 계산하는 식은 다음과 같다.

모든 테스트문서에 대한 임시 문서분류함수들 간의 변이

$$= \sum_{x \in X_T} P(x) variance_x \quad (11)$$

$$variance_x \equiv \frac{1}{2}\left(1 - \sum_{y \in Y} [P(Y_H = y \mid x)]^2\right) \quad (12)$$

$$P(Y_H = y \mid x) \equiv \frac{\left(\sum_{t=1}^{n} P(Y_{h_t} = y \mid x)\right)}{n} \quad (13)$$

식 (13)에서 는 특정 임시 문서분류함수가 문서 x를 카테고리 y로 분류했는지 여부를 나타낸다. 문서 x를 카테고리 y로 분류했으면 이 값은 1 이고 그렇지 않다면 0 의 값을 갖는다. 이에 대하여 모든 임시 문서분류함수에 대하여 평균을 낸 값이 이다. 특정 문서 x에 대한 임시 문서분류함수들의 변이는 식 (12)와 같이 정의되는데, 결국 각 임시 문서분류함수가 문서 x를 서로 다르게 분류할수록 변이의 값은 커진다. 그림 11은 임시 문서분류함수 h_1, h_2, h_3 간의 변이를 테스트문서 x_1, x_2, x_3에 대하여 측정하는 예를 보여준다.

Hypothesis	h_1			h_2			h_3			Variance
카테고리 / 문서	y_1	y_2	y_3	y_1	y_2	y_3	y_1	y_2	y_3	
x_1	1	0	0	1	0	0	1	0	0	$Variance_{x1}$ =1-(1^2+0^2+0^2)=0
x_2	1	0	0	1	0	0	0	1	0	$Variance_{x2}$=0.4555
x_3	1	0	0	0	1	0	0	0	1	$Variance_{x3}$=0.6733
Total Variance = (1/3)*0 + (1/3)*0.4555 + (1/3)*0.6733 = 0.3762										

그림 11 임시 문서분류함수들 간의 변이 측정 예

4.3 AdaBUS 알고리즘

그림 12는 AdaBUS 알고리즘을 의사코드로 나타낸 것이다. AdaBUS 알고리즘은 입력으로 초기 학습문서집합인 D_T와 라벨이 없는 문서집합인 X_U를 받는다(1행). 초기에는 학습문서집합에 있는 모든 학습문서의 비중을 서로 같게 초기화 한다(2행). 그 후 알고리즘은 샘플링하려는 학습문서의 개수만큼 루프를 진행하면서 같은 수만큼의 임시 문서분류함수를 만든다. 부스팅 알고리즘이 고정된 수인 T만큼 루프를 진행했던 것과는 비교된다.

루프를 m번 진행시키는 동안 알고리즘은 샘플링과 비중 재조정 작업을 통해 m개의 임시 문서분류함수를 만든다. 우선 현재의 학습문서집합과 그 비중을 이용하

여 임시 문서분류함수를 만들고, 이를 통해 에러와 신뢰도를 계산한다(3-a,b,c행). 그 후 만든 임시 문서분류함수를 이용하여 X_U로부터 불확실성이 가장 큰 문서를 하나 골라서 학습문서로 채택한다(3-d,e행). 불확실성 기반 샘플링 알고리즘을 이용하여 선택한 학습문서의 비중은 모든 학습문서들의 비중의 평균으로 초기화 하고, 부스팅 알고리즘의 전략에 따라 학습문서집합에 속한 각 학습문서들의 비중을 다시 조정한다(3-f행). 마지막으로 4행에서는 루프를 진행하면서 만들어낸 m개의 임시 문서분류함수를 이용하여 투표방법을 통해 최종 문서분류함수를 만들어낸다.

1. 입력: 초기 학습문서집합 $D_T=\{ <x_1, c(x_1)>, <x_2, c(x_2)>, \cdots, <x_n, c(x_n)> \}$,
   라벨이 없는 문서집합 X_U
2. 모든 $x \in D_T$에 대하여, $W_1(x) = 1/n$
3. For $t=1, \cdots, m$
   3-a) D_T와 이들의 비중 분포인 W_t을 이용하여 임시 문서분류함수인 h_t를 만든다.
   3-b) h_t의 에러를 다음과 같이 구한다.

$$\varepsilon_t = \sum_{x_i \in D_T} W_t(x_i)\delta(x_i), \quad \delta(x_i) = \begin{cases} 0, & if\ h_t(x_i) = c(x_i) \\ 1, & if\ h_t(x_i) \neq c(x_i) \end{cases}$$

   3-c) h_t의 신뢰도인 $\alpha_t = \frac{1}{2}\ln(\frac{1-\varepsilon_t}{\varepsilon_t})$를 구한다.
   3-d) h_t를 이용하여, X_U로부터 가장 불확실성이 큰 문서인 x를 선택한다.
   $x \leftarrow U(X_U)$
   3-e) 전문가가 x의 라벨을 부여한 후, 이를 학습문서집합에 추가한다.
   $D_T \leftarrow D_T \cup <x, c(x)>$
   3-f) 비중 재조정:
   $W_1(x)=1/|D_1|$
   모든 $x_i \in D_T$에 대하여,

$$W_{t+1}(i) = \frac{W_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t}, & if\ h_t(x_i) = c(x_i) \\ e^{\alpha_t}, & if\ h_t(x_i) \neq c(x_i) \end{cases}$$

   여기서 Z_t 는 가 분포가 되도록 하기위한 정규화인수이다.
4. 투표를 이용하여 최종 문서분류함수를 만든다:
   $h(x) = voting(<h_1, a_1, x>, \cdots, <h_m, a_m, x>)$

그림 12 AdaBUS 알고리즘의 의사코드

5. 실 험

5.1 실험 설계

5.1.1 데이타 집합

본 연구에서는 문서분류시스템을 구성하고 이것의 분류 정확도를 측정하기 위해서 Reuter-21578 문서집합을 이용하였다. 이 문서집합은 일반적으로 문서분류시스템을 평가하기 위한 실험에서 많이 사용된다[18]. 표 1은 실험에서 쓰인 카테고리와 각 카테고리에 속하는 문서집합의 용도를 설명한 것이다.

표 1 실험을 위한 데이타 집합

카테고리	전체 문서집합 원소개수	초기 학습 문서집합 원소개수	테스트 문서집합 원소개수	라벨이 없는 문서집합 원소개수
acq	264	5	30	229
crude	165	5	30	130
earn	504	5	30	469
interest	249	5	30	214
money-fx	179	5	30	144
ship	222	5	30	187
trade	150	5	30	115

표 1에서 볼 수 있듯이 전체 7개의 카테고리가 실험에 쓰였으며, 각 카테고리에서 5개의 문서를 임의로 선택하여 총 35개의 초기 학습문서집합을 구성하였다. 그리고 각 카테고리로부터 30개의 문서를 임의로 선택하여 총 210개의 테스트 문서집합을 구성하였고, 나머지 1453개의 문서로는 라벨이 없는 문서집합을 구성하였다. 실험의 공정성을 위하여 이러한 데이타 집합 구성을 임의로 10회 구성하여 실험을 실시하였다.

5.1.2 속성집합선택(feature selection)

일반적으로 문서 분류의 성능을 높이고 분류 계산시간을 줄이기 위하여 본 실험에서는 속성집합선택을 수행한다. [19]에서는 이것을 위한 대표적인 방법인 문서빈도(document frequency), 정보이득량(information gain), 카이제곱통계량(-statistics), 상호정보량(mutual information) 그리고 용어강도(term strength)를 기준값으로 한 방법을 소개하였고, 그것들의 성능을 비교하였다. 본 실험에서는 [19]의 실험 결과를 반영하여 시간적으로 효율성이 좋으면서 분류성능을 높이는 그 결과 앞의 세 방법이 상대적으로 효과적임을 밝혔다. 여기서는 문서빈도를 기준값으로 한 속성선택 기법을 사용 방법을 사용하였다. 앞에서 소개한 5가지 속성집합선택 방법들을 Naïve Bayes 문서 분류기에 적용한 결과, [32]에서와 같이 앞의 세 방법의 성능이 비슷하게 효과적인 것으로 밝혀졌다. 특정 단어의 문서빈도는 해당 단어가 출현하는 학습 문서의 수를 의미한다. 희귀한 단어는 문서 분류를 하는데 있어 정보를 거의 제공하지 못한다는 것이 이 방법의 기본적인 가정이므로, 이 방법은 문서빈도가 높은 것을 우선하여 속성으로 선택한다. 본 논문에서는 학습 문서 집합 내의 각 단어들에 대하여 문서빈도를 계산한 후에, 이 수치로 단어들의 순위를 부여하여 상위 30%인 것들만 속성으로 선택하였다.

5.1.3 성능 평가 측정치

여러 개의 카테고리 중 하나의 카테고리에 입력 문서를 할당하는 문서분류의 문제에서 각 카테고리와 각 테

스트 문서에 대하여 특정 문서가 특정 카테고리로 분류되었는지 여부에 따라 다음과 같은 측정치를 계산할 수 있다[18].

- *a*: 해당 카테고리에 정확하게 분류된 문서의 수(*true positive*)
- *b*: 해당 카테고리에 틀리게 분류된 문서의 수(*false positive*)
- *c*: 해당 카테고리에 속하지만 이 카테고리로 분류되지 않은 문서의 수 (*false negative*)
- *d*: 해당 카테고리에 속하지 않고, 이 카테고리로 분류되지 않은 문서의 수 (*true negative*)

이를 합쳐서 해석하면, $a+c$는 해당 카테고리에 속하는 모든 문서의 수이고, $a+b$는 문서분류함수에 의해 해당 카테고리에 실제로 분류된 문서의 수이다. 이를 통해서 이제 *recall*과 *precision*을 정의해보자.

$$recall = \frac{a}{a+c} \tag{14}$$

$$precision = \frac{a}{a+b} \tag{15}$$

*recall*과 *precision*은 일반적으로 문서분류시스템의 성능을 평가하는 측정치로서 많이 사용된다. 이 두 측정치 모두 문서분류시스템을 평가하는데 있어 매우 중요하기 때문에 이 둘의 중요성을 모두 반영한 측정치가 필요한데, 생각할 수 있는데 이를 위해 F_1 측정치를 사용한다. 다음은 F_1 측정치의 표현식이다.

$$F_1 = \frac{2*recall*precision}{recall+presision} \tag{16}$$

위에서 설명한 *recall*과 *precision*, F_1 측정치는 각 카테고리의 성능을 개별적으로 평가하는 것이다. 모든 카테고리에 대한 평균적인 성능을 평가하기 위해 여기서는 *macro-averaging* 방법을 이용한다. 이 방식에서는 각 카테고리 별로 *recall*, *precision*, F_1 측정치 등을 계산하고 이들의 평균을 계산하여 전체적인 문서분류시스템의 성능을 평가한다.

5.2 실험결과 및 분석

그림 13은 각 알고리즘의 실험 결과를 나타낸 것이다. X축은 초기 학습문서집합에 추가된 학습문서의 수를 의미하고, Y축은 주어진 학습문서집합을 이용한 각 알고리즘의 F_1 측정치를 나타낸다. 따라서 그래프가 좀 더 위에 위치할수록 알고리즘의 문서분류 정확도가 높다는 것을 의미한다. 그림에서 US는 불확실성 기반 샘플링을 의미하고, RS는 무작위 샘플링(random sampling)을 의미한다. 무작위 샘플링이란 라벨이 없는 문서집합으로부터 임의로 문서를 선택하여 학습문서를 추가하는 방법을 말한다. 이제 각 알고리즘의 실험 결과를 차례대로 분석해보자.

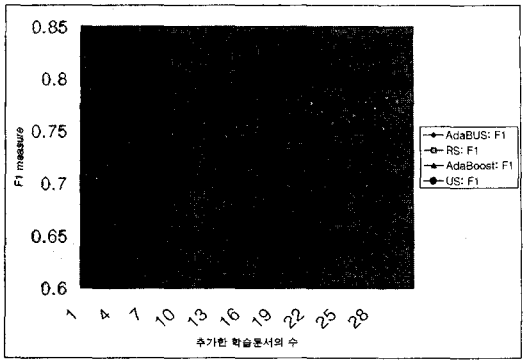

그림 13 추가한 학습문서의 수에 따른 각 알고리즘의 F_1 측정치

AdaBoost 알고리즘의 실험 결과를 살펴보자. 이 알고리즘의 문서분류 정확도는 예상 대로 좋지 않다. 2.2절에서 언급했듯이, Naïve Bayes 문서분류 알고리즘은 상당히 안정적인 알고리즘이기 때문에 AdaBoost 알고리즘을 적용해도 교란의 효과를 얻을 수가 없다. 이러한 이유 때문에 그림에서 보듯이 AdaBoost 알고리즘을 적용하지 않은 무작위 샘플링의 결과보다 AdaBoost 알고리즘을 적용한 문서분류 알고리즘의 결과가 좋지 않다.

이제 불확실성 기반 샘플링 알고리즘의 결과를 살펴보자. 이 알고리즘은 무작위 샘플링보다 분류정확도 면에서 우수하다. 그 이유는 불확실성 기반 샘플링 알고리즘을 이용해서 선택한 문서들이 임의로 선택한 문서들보다 훨씬 정보량이 크기 때문으로 분석된다[4]. 불확실성 기반 샘플링 알고리즘을 적용함으로써 보다 적은 학습문서를 통해 보다 정확한 문서분류함수를 얻을 수 있다는 사실을 실험결과를 통해 확인할 수 있다.

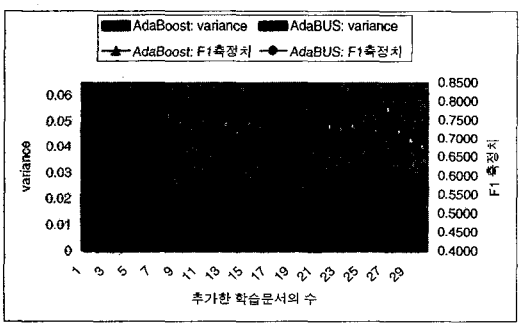

그림 14 임시 문서분류함수들 간의 변이 측면에서 살펴본 AdaBoost와 AdaBUS의 문서분류 정확도 비교

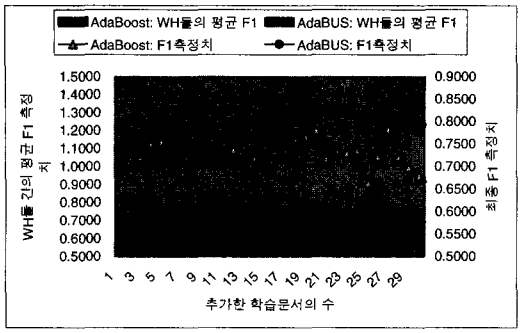

그림 15 임시 문서분류함수들의 평균 F1 측정치 측면에서 살펴본 AdaBoost와 AdaBUS의 문서분류 정확도 비교

마지막으로 AdaBUS 알고리즘의 실험결과를 살펴보자. 그림 13에서 볼 수 있듯이, AdaBUS 알고리즘의 문서분류 정확도는 모든 알고리즘 중 가장 높다. 그 이유는 4.1절에서도 언급했듯이, AdaBUS 알고리즘이 AdaBoost 알고리즘의 장점과 불확실성 기반 샘플링 알고리즘의 장점을 모두 수용했기 때문으로 분석된다. 불확실성 기반 샘플링 알고리즘을 진행하면서 임시 문서분류함수들을 만들었기 때문에 이들 간의 교란이 보다 잘 이루어졌으며 따라서 이들의 투표 결과로 만들어진 최종 문서분류함수의 정확도가 높은 것이다.

그림 14는 4.2절에서 정의한 임시 문서분류함수들 간의 변이 측정치를 이용하여, AdaBoost 알고리즘과 AdaBUS 알고리즘이 만들어내는 임시 문서분류함수들 간의 변이를 측정한 것이다. 그림에서 볼 수 있듯이 항상 AdaBUS가 만들어 내는 임시 문서분류함수들 간의 변이가 더 크다는 것을 알 수 있다. 이를 통해 AdaBUS 알고리즘은 Naïve Bayes 문서분류 알고리즘이 갖고 있는 안정적인 특성을 극복한 것을 확인할 수 있다.

그림 15는 AdaBoost 알고리즘과 AdaBUS 알고리즘이 만들어 내는 임시 문서분류함수들의 평균 F_1 측정치를 보여준다. 대부분의 경우 AdaBUS 알고리즘의 임시 문서분류함수들의 평균 F_1 측정치가 크다는 것을 알 수 있다. 임시 문서분류함수들 간의 변이와 이들의 평균 F_1 측정치면에서 AdaBUS 알고리즘이 AdaBoost 알고리즘보다 크기 때문에 최종 문서분류함수의 정확도면에서 AdaBUS 알고리즘이 AdaBoost 알고리즘보다 높은 것으로 분석된다.

6. 결 론

문서분류함수를 추정하는 알고리즘과 학습문서집합의 구성방법은 기계학습 기법을 이용한 문서분류시스템의 분류 정확도를 결정하는 가장 중요한 두 가지 요인이다. 본 논문에서는 문서분류함수 추정 알고리즘인 Naïve Bayes 문서분류 알고리즘에 기존의 학습문서집합 구성방법을 적용해본 후에 그들의 장단점을 분석하여 AdaBUS 알고리즘이라는 새로운 학습문서집합 구성방법을 제안하였다. 그리고 실험을 통해 AdaBUS 알고리즘을 적용한 Naïve Bayes 문서분류시스템의 정확도가 다른 알고리즘을 적용한 것에 비해 높다는 사실을 입증하였다. AdaBUS 알고리즘을 이용한 문서분류시스템은 보다 적은 수의 학습문서로 보다 높은 정확도의 문서분류 결과를 얻을 수 있었다.

향후 연구과제로는 AdaBUS 알고리즘의 우수성을 수학적으로 분석해보는 방안을 생각할 수 있다. 본 논문에서는 AdaBUS 알고리즘이 다른 알고리즘에 비해 문서분류함수의 정확도를 높여준다는 사실을 경험적인 실험에 의해 증명하였는데, 향후 이를 수학적으로 입증한다면 AdaBUS 알고리즘의 우수성을 보다 공정하고 정확하게 평가할 수 있을 것이다. 또한 본 논문에서는 AdaBUS 알고리즘을 Naïve Bayes 문서분류 알고리즘에 적용하였지만 이를 최근 들어 문서분류시스템을 구성하는데 자주 사용되고 있는 SVM 알고리즘 등에 적용해 보는 것도 흥미로운 작업이 될 것이다.

참 고 문 헌

[1] Tom M. Mitchell. Machine Learning. *McGraw-Hill International Editions*, chapter 6, 1997.

[2] R. Agrawal, R. Bayardo, and R. Srikant. Athena: Mining-based Interactive Management of Text Databases. *In Proceedings of the 7th International Conference on Extending Database Technology*, pages 365-379, 2000.

[3] Pedro Domingos and Michael Pazzani. Beyond Independence: Conditions for the Optimality of the Simple Bayesian Classifier. In *Proceedings of the 13th International Conference on Machine Learning*, pages 105-112, 1996.

[4] 김제욱, 김한준, 이상구. Naïve Bayes 문서 분류기를 위한 점진적 학습 모델 연구. *정보기술과 데이타베이스 저널*, 8(1), pages 95-104, 2001.

[5] David D. Lewis and William A. Gale. A Sequential Algorithm for Training Text Classifiers. In *Proceedings of the 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 3-12, 1994.

[6] Yoav Freund and Robert E. Schapire. Experiments with a New Boosting Algorithm. In *Proceedings of the 13th International Conference on Machine*

Learning, pages 148-156, 1996.

[7] David D. Lewis and Jason Catlett. Heterogeneous Uncertainty Sampling for Supervised Learning. In *Proceedings of the 11th international Conference on Machine Learning*, pages 148-156, 1994.

[8] M. Trensh, N. Palmer, and A. Luniewski. Type Classification of Semi-structured Documents. *In Proceedings of the 21st ACM SIGMOD International Conference on Management of Data*, 1995.

[9] Yoav Freund and Robert E. Schapire. A Decision-theoretic Generalization of On-line Learning and an Application to Boosting. *Journal of Computer and System Sciences*, 55(1), pages 119-139, 1997.

[10] J. R. Quinlan. Bagging, Boosting, and c4.5. *In Proceedings of the13th National Conference on Artificial Intelligence*, pages 725-730. 1996.

[11] Robert E. Schapire. The Strength of Weak Learnability. *Machine Learning*, 5(2), pages 197-227, 1990.

[12] Robert E. Schapire and Yoram Singer. Boos Texter: A Boosting-based System for Text Categorization. *Machine Learning*, 39(2), pages 135-168, 2000.

[13] Robert E. Schapire and Yoram Singer. Improved Boosting Algorithms Using Confidence-rated Predictions. *Machine Learning*, 37(3), pages 297-336, 1999.

[14] Leo Breiman. Arcing Classifiers. *The Annals of Statistics*, 26(3), pages 801-849, 1998.

[15] Kai Ming Ting and Zijian Zheng. Improving the Performance of Boosting for Naïve Bayesian Classification. *In Proceedings of the 3rd Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 1999.

[16] Zijian Zheng. Naïve Bayesian Classifier Committees. In *Proceedings of European Conference on Machine Learning*, pages 196-207, 1998.

[17] Ron Kohavi, David H. Wolpert. Bias Plus Variance Decomposition for Zero-One Loss Functions. *In Proceedings of the 13th International Conference on Machine Learning*, pages 275-283, 1996

[18] Yiming Yang. An Evaluation of Statistical Approaches to Text Categorization. *Journal of Information Retrieval*, 1(1), pages 67-88, 1999.

[19] Yiming Yang and J. O. Pedersen. A Comparative Study on Feature Selection in Text Categorization. In *Proceedings of the 14th International Conference on Machine Learning*, pages 42-420, 1997.

김 제 욱

1999년 서울대학교 해양학과 졸업(학사). 2002년 서울대학교 컴퓨터공학부 졸업(석사). 2002년 1월 ~ 현재 대우정보시스템 기술연구소. 관심분야는 데이타마이닝, 전자상거래, 웹 서비스

김 한 준

1994년 서울대학교 계산통계학과 졸업(학사). 1996년 서울대학교 전산과학과 졸업(석사). 2002년 서울대학교 컴퓨터공학부 졸업(박사). 2002년 9월 ~ 현재 서울대학교 컴퓨터공학부 BK21연수연구원. 관심분야는 데이타베이스, 데이타마이닝, 텍스트마이닝, 지식관리, 정보검색, 지능형 정보시스템

이 상 구

1985년 서울대학교 계산통계학과 졸업(학사). 1987년 Northwestern University 졸업(석사). 1990년 Northwestern University 졸업(박사). 1989년 9월 ~ 1990년 6월 University of Minnesota 전임강사. 1990년 ~ 1992년 7월 Electronic Data Systems 연구원. 1992년 8월 ~ 현재 서울대학교 컴퓨터공학부 부교수. 관심분야는 논리데이타베이스, 정보검색, 전자상거래 기술, 전자도서관