정보검색 기술을 이용한 비지도 학습 기반 문서 분류 시스템 개발

(Developing a Text Categorization System Based on Unsupervised Learning Using an Information Retrieval Technique)

노 대 욱 † 이 수 용 † 나 동 열 ††
(Dae-Wook Noh) (Soo-Yong Lee) (Dong-Yul Ra)

요 약 문서분류기의 개발에 있어 지도학습기법을 이용할 경우 많은 양의 사람에 의한 범주 부착 말뭉치가 필요하다. 그러나 이의 구축은 많은 시간과 노력을 필요로 한다. 최근 이러한 범주 부착 말뭉치 대신 원시말뭉치와 범주마다 약간의 씨앗 정보를 이용하여 학습을 수행하여 문서분류기를 개발하는 방법론이 제시되었다. 본 논문에서는 이 방법론 하에서 다른 연구에서의 결과보다 좋은 성능을 나타내는 비지도 학습 기법을 소개한다. 본 논문에서 제시하는 기법의 특징은 씨앗 단어에서 출발하여 평균상호정보를 이용하여 다른 대표단어 및 그들의 가중치를 학습한 다음, 정보검색에서 많이 사용하는 기술을 이용하여 그 가중치를 갱신하는 것이다. 그리고 이 과정을 반복 수행하여 최종적으로 높은 성능의 시스템을 개발 할 수 있음을 제시하였다.

키워드 : 문서분류, 비지도학습, 대표단어, 상호정보, 정보검색

Abstract for developing a text classifier using supervised learning, a manually labeled corpus of large size is required. However, it takes a lot of time and human effort. Recently a research paradigm was proposed to use a raw corpus and a small amount of seed information instead of manually labeled corpus. In this paper we introduce an unsupervised learning method that makes it possible to achieve better performance than other related works. The characteristics of our approach is that average mutual information is used to learn representative words and their weights and then update of the weights is done using a technique inspired by the works in information retrieval. By iterating this learning process it was shown that a high performance system can be developed.

Key words : Text classification, unsupervised learning, representative words, mutual information, information retrieval

1. 서 론

문서의 자동분류 기술은 문서의 내용에 기반 하여 미리 정의된 범주(category)로 문서들을 분류하는 것을 말한다. 다뤄야 하는 문서의 수가 너무 많아 일일이 모든 문서를 읽어 보기 어려울 정도로 많은 양의 문서가 매일 밀려드는 현대 사회에서 문서 자동 분류 시스템은 그 중요성이 날로 커지고 있다.

이러한 문서 분류기의 개발에는 주로 기계 학습 방법을 사용한다[1]. 특히 이미 범주가 부착된 말뭉치를 학습 데이타로 이용하는 지도학습 기법을 이용하는 것이 가장 일반적인 방법이다. 기존의 지도 학습(supervised learning) 알고리즘을 이용한 연구들을 살펴보면 문서 분류기의 성능이 매우 높은 것을 확인할 수 있다[2,3]. 하지만 이런 높은 성능에 이르기 위해서는 사람이 수작업으로 범주 레이블(label)을 붙인 충분한 양의 학습 말뭉치가 미리 준비되어 있어야 한다. 그러나 사람들이 직접 레이블을 붙여 놓은 학습 말뭉치를 준비하기 위해서는 많은 시간과 노력이 필요한 일이므로 쉬운 일이 아니다.

이러한 수동 태깅 말뭉치 구축의 어려움을 피하고 자 하는 연구가 시도되었는데 그 기본 아이디어는 비지도 학습 기법을 사용하는 것이다. 그러나 오직 원시 말뭉치만을 이용하는 완전한 비지도학습 만을 사용하는 경우

† 비 회 원 : 연세대학교 정보통신공학부
dwnoh2272@dragon.yonsei.ac.kr
hosu@hosu.yonsei.ac.kr
†† 종신회원 : 연세대학교 정보통신공학부 교수
dyra@yonsei.ac.kr
논문접수 : 2006년 8월 13일
심사완료 : 2006년 8월 31일

에는 높은 성능의 문서분류 시스템을 개발하는 것이 어렵다는 것이 알려 졌다. 따라서 원시 말뭉치 이외에 약간의 추가적인 씨앗 정보를 주어 학습을 시키는 방법이 제안되었다. 씨앗 정보로는 아주 작은 양의 수동 태깅 말뭉치를 제공하는 것을 시도하였다[4,5]. 이들 연구에서는 원시 말뭉치를 이용하기 위해 Expectation Maximization 알고리즘을 응용하는 기법을 사용하였다.

이와 같은 방법론을 사용하지만 씨앗 정보로 수동 태깅 말뭉치 대신에 범주를 대표할 수 있는 몇 개의 씨앗 단어를 제공하자는 대한 연구들이 최근에 시도되었다[6-8]. 시스템이 이용할 수 있는 초기 입력 정보는 범주 레이블이 없는 문서(원시 말뭉치)들과 각 범주를 대표하는 씨앗(seed) 단어들이다. 이러한 입력 데이타를 이용하여 먼저 비지도 학습 기법을 통하여 각 범주에 대한 정보를 학습한다. 그리고 이를 이용하여 원시 말뭉치의 각 문서를 분류하여 범주 레이블을 부착한다. 이 결과로 기계-표지-부착 말뭉치(machine-labeled corpus; MLC)를 얻는다. 이렇게 얻은 MLC를 이용하여 지도학습 알고리즘을 학습하여 최종의 문서 분류기를 얻게 된다. 즉 주어진 방법론에서는 첫 단계로 비지도 학습 알고리즘을 이용하고, 다음 단계로 지도 학습 알고리즘을 이용하는 단계를 거친다. 본 논문의 연구도 이 방법론을 따른다. 이러한 방법론을 따르는 연구들 사이의 주된 차이점은 첫 단계의 비지도 학습 기법이다. 두 번째 단계의 지도 학습은 이미 잘 알려진 표준화된 지도학습 알고리즘을 그대로 적용하는 것에 불과하여 큰 차이가 없다. 전체 시스템의 성능에 가장 큰 영향을 미치는 것은 비지도 학습의 결과인 MLC의 질(quality)이다. 비지도 학습 기법은 연구마다 다른 기법을 취할 여지가 많고 변화의 여지가 많다.

이에 따라 본 논문에서는 새로운 비지도 학습기법을 소개한다. 본 논문의 연구에서는 정보검색에서 각 문서가 벡터로 표현되는 것처럼 각 범주들을 문서와 같이 벡터로 가능한 한 정확하게 나타내고(이를 범주 대표 벡터라 함) 이 대표 벡터들을 사용하여 정보검색에서처럼 cosine 계수 등과 같은 비교 기법을 이용하여 문서를 분류하는 것이다. 그리고 이를 이용하여 MLC를 생성한다.

본 논문의 첫째 특징은 비지도 학습 단계에서 평균 상호정보(average mutual information) 개념을 사용한 것이다. 이를 이용하여 씨앗 단어에서 출발하여 범주에 대한 대표성이 있는 단어들과 그 가중치를 수집한다. 둘째 특징은 이렇게 하여 만들어진 범주 대표 벡터를 이용하여 문서를 분류한 다음 그 결과를 이용하여 각 대표 단어들에 대한 보다 더 정확한 가중치를 구하는 단계(가중치 갱신 단계)를 두었다는 점이다. 이 가중치 갱신을 위해서 정보검색에서 자주 사용되는 tf-idf 개념을 이용하였다[9]. 그 결과 보다 정확한 대표 벡터를 얻게 함으로써 문서 분류 성능을 대폭 향상시키는 것을 관찰하였다. 세째 특징은 이 과정을 반복시킴으로써 더욱 성능이 향상되도록 한 점이다. 실험을 통하여 본 논문에서 제안한 비지도 학습 기법이 다른 연구에서 제안한 비지도 학습 기법보다 우수한 성능을 나타냄을 확인할 수 있었다.

본 논문의 구성은 다음과 같다. 2장에서는 문서분류 개발 시스템의 전체적인 구성과 그 배경 정보를 설명한다. 3장에서는 본 연구에서 개발한 비지도 학습 기법에 대하여 살펴본다. 4장에서는 지도학습에 의한 최종 문서 분류기의 구축에 대하여 기술한다. 개발된 시스템에 대한 실험 및 검토를 5장에서 다루고 마지막으로 6장에서 결론을 제시한다.

2. 전체 구성

우리 시스템은 그림 1과 같이 학습 과정에서 순차적으로 적용되는 두 개의 모듈로 구성되어 있다. 모듈 1에서는 레이블이 없는 원시 문서들을 이용하여 비지도 학습을 수행한 후 이들 문서에 범주를 결정하여 붙이게 된다. 첫 모듈은 레이블이 없는 문서 즉 원시 문서들만 사용하게 되므로 비지도 학습 방법이 된다. 그러나 초기에 아무런 정보도 제공 되지 않으면 높은 성능을 기대하기 어렵기 때문에 초기에 입력 정보로서 씨앗(seed) 단어를 제공한다. 이러한 씨앗 단어는 각 범주 이름을 구성하고 있는 단어만을 사용한다[6,7]. 예를 들어 20 Newsgroup 데이타셋(dataset)의 한 범주인 "rec.sport.baseball"에 대해서는 씨앗 단어로 공통된 범주명인 rec와 sport를 제외한 baseball을 사용하게 된다.

모듈 2에서는 지도 학습 방법으로 문서 분류기를 얻는다. 지도 학습을 위해서는 보통 사람이 붙인 정답 레이블을 가진 말뭉치를 이용한다. 그러나 본 시스템에서는 이러한 정답 말뭉치가 존재하지 않으므로 모듈 2가 사용하는 학습 문서 집단은 모듈 1의 결과인 기계가 범주를 붙여준 문서 집단 즉 MLC가 된다. 모듈 2는 대부

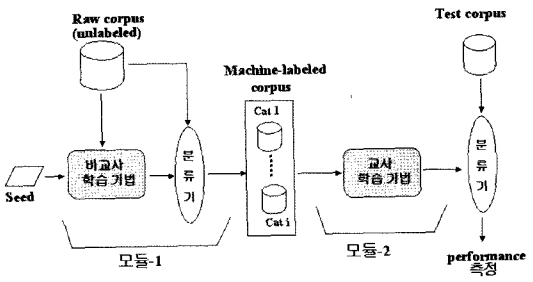

그림 1 비지도 학습 기반 문서 분류기 개발 시스템

분의 기존 연구에서와 같이 문서 분류에서 좋은 성능을 보이는 어떠한 지도학습 알고리즘을 사용하여도 무방하다. 우리는 최근에 가장 많이 이용되고 있는 지도학습 알고리즘인 Support Vector Machine(SVM)을 사용하였다[6]. 모듈 2의 결과로 얻는 문서 분류 시스템이 우리가 얻는 최종적인 문서 분류기이다.

문서 분류 시스템의 개발에 있어 먼저 결정하여야 할 것은 자질 집합(feature set)으로서 문서의 경우 자질은 일반적으로 색인어 또는 키워드가 된다. 색인어로 문서에 나타나는 모든 단어를 이용하는 경우 그 수가 너무 크기 때문에 과도한 저장공간 및 계산량을 필요로 할 수 있어서 그 수를 적정 수준 이하로 낮추어야 한다. 이러한 작업을 자질선택(feature selection) 작업이라고 한다[10]. 본 논문에서는 그러나 특별한 자질 선택 기법을 사용하지 않고 매우 간단한 방법을 취하였다. 우리는 말뭉치에 나타난 모든 단어들 중에서 품사가 동사, 명사, 고유명사, 형용사 만을 고려하고 나머지는 제거한다. 우리가 다루는 문서 집단의 경우 사람이나 기관의 이름 등 미등록어가 많이 발생하였는데 이들도 색인어 대상에 포함시킨다. 명사, 동사의 경우 원형 복원(stemming) 과정을 거쳐 구한 원형을 이용한다. 일반적인 문서 분류기에서는 자질의 수를 줄이기 위하여 자질 선택(feature selection)과정을 거친다. 우리의 경우는 말뭉치에서 단어가 나타난 전체 발생 횟수(total term frequency)가 특정 횟수 이하인 경우와 단어가 나타난 문서의 수(document frequency)가 특정 수 이상인 것들을 제거하는 간단한 방법만을 사용하였다.

정보검색에서 사용하는 벡터 공간 모델에서 문서와 질의가 벡터로 표시되는 방식과 동일하게, 각 범주는 색인어 집합 V 안의 각 대응되는 단어의 가중치를 원소로 갖는 벡터로 표현된다. 색인어 집합 V는 순서화된(ordered) 집합으로서 시스템이 문서나 범주를 표현하는데 이용되는 모든 색인어로 이용되는 단어를 포함한다(|V|는 전체 색인어의 수):

$$V = \{w_1, \cdots, w_i, \cdots, w_{|V|}\} \quad (1)$$

시스템은 또한 미리 정해진 범주를 가지고 있다고 가정한다. 각 범주는 범주 레이블로 나타내지나 순서화된 범주 집합 C 안에서 이 범주 레이블이 차지하는 위치로 나타낼 수 도 있다(|C|는 전체 범주의 수) :

$$C = \{l_1, \cdots, l_c, \cdots, l_{|C|}\} \quad (2)$$

모듈 1의 비지도 학습 기법을 개발하는 데 있어서 우리는 정보검색적인 관점에서 접근한다. 우리는 문서 분류 문제를 정보검색의 문제와 대응하여 생각하도록 한다. 정보검색에서의 질의를 문서 분류에서의 분류할 문서로 대응시키고, 정보검색에서의 문서집단 내의 문서를 문서분류에서의 범주로 대응시킨다(그림 2 참조). 결국 정보검색에서 질의에 가장 적합한 문서를 찾는 문제를 문서분류에서는 분류할 문서에 가장 적합한 범주를 찾는 문제로 대응시킬 수 있다. 그리고 질의, 문서, 범주를 모두 벡터로 나타내었으므로 정보검색에서의 기술을 그대로 문서분류에 이용할 수 있다.

정보검색에서 사용하는 벡터 공간 모델에서 문서와 질의가 벡터로 표시되는 방식과 동일하게, 각 범주는 색인어 집합 V안의 각 대응되는 단어의 가중치를 원소로 갖는 대표벡터로 표현된다. 예를 들어 c 번째 범주는 대표 벡터 R_c로 나타낸다.

$$R_c = < u_{c,1}, u_{c,2}, \cdots, u_{c,|V|} > \quad (3)$$

문서도 다음과 같이 같은 원소 수를 갖는 벡터로 나타내어진다.

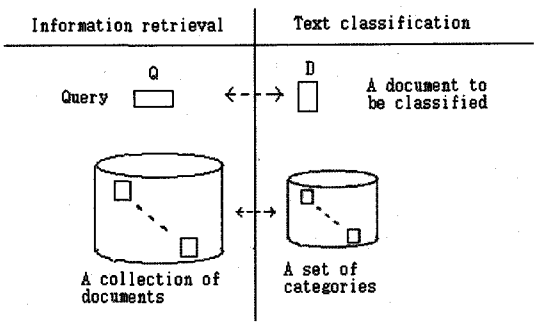

그림 2 정보검색과 문서분류의 대응관계

$$D = < a_1, a_2, \cdots, a_{|V|} > \quad (4)$$

각 범주에 대한 대표 벡터가 모두 마련되어 있으면 분류할 임의의 문서 D의 범주를 결정할 수 있다. 이를 위해 문서 D와 문서 R_c의 유사도를 이용하는 것이다. 이는 정보검색에서 효율적으로 사용되는 cosine 값을 이용해 계산한다[9].

$$sim(D, R_c) = \cos ine(D, R_c) \quad (5)$$

주어진 문서와 모든 범주에 대한 대표벡터 사이의 유사도가 계산된 후 이 문서에 대한 범주는 다음과 같이 결정할 수 있다.

$$Category_{chosen} = ARGMAX_c sim(D, R_c) \quad (6)$$

그렇다면 결국 우리의 문제는 어떻게 범주 대표벡터를 구할 것인가에 있다. 범주에 대한 대표벡터를 구하기 위하여 우리는 범주에 대한 대표 단어 개념을 도입한다. 범주 대표 단어란 해당 범주를 잘 나타낼 만한 단어이다. 물론 색인어 집합 V 안의 모든 단어를 대표단어로 보고 각 단어의 중요도를 잘 결정하도록 하는 방식을 취할 수도 있다.

그러나 우리는 학습의 편리성을 위하여 각 범주마다 그 범주를 대표할 만한 단어들을 학습하도록 하였다. 이런 단어를 범주 대표 단어라 부른다. 한 범주에 대해 범주 대표 단어의 수는 수백 개로서 전체 단어 수 |V| 보다는 훨씬 작도록 한다. 결국 비지도 학습의 일차적인 목표는 범주 대표 단어들을 학습하는 문제이다. 그리고 대표 단어 들 사이에도 중요도의 차이가 있으므로 그 중요도를 나타내는 값인 대표 단어의 가중치도 학습하도록 한다.

대표 단어를 학습한 후에는 이를 이용하여 대표 벡터를 마련할 수 있다. 만약 V의 j 번째 단어 w_j가 범주 c의 대표 단어로 학습되었다면 대표벡터 R_c의 대응되는 원소 u_{cj}를 w_j의 그 가중치로 놓도록 한다. 범주 c의 대표단어가 되지 못한 모든 단어들에 해당하는 원소에는 0을 넣도록 한다. u_{cj}의 값은 w_j가 얼마만큼 범주 c를 대표하는 대표성이 있는 지를 나타내는 가중치이다. 각 범주에 대한 대표단어의 학습은 다음 장에서 다룬다.

이와 같이 모듈 1에서 최종적으로 구한 범주 대표벡터로 구성한 분류기를 사용하여 원시 말뭉치의 모든 문서에 범주를 부착한다. 그 결과인 기계에 의한 범주 부착 말뭉치, MLC, 는 모듈 2에게 전달된다. 모듈 2는 이 MLC를 훈련용 말뭉치로 이용하여 지도학습 알고리즘을 훈련시켜서 최종적인 문서분류 시스템을 구축한다(그림 1 참조).

3. 범주 대표 벡터의 비지도 학습

본 장에서는 모듈 1에서 수행하는 비지도 학습 기법에 대해 살펴 본다. 이는 스텝 1과 스텝 2의 두 단계로 구성되며(그림 3 참조), 이 두 단계는 루프를 형성하여 여러 번 반복 수행된다.

3.1 평균 상호 정보를 이용한 대표 단어 학습 및 가중치 결정

본 절에서는 비지도 학습의 두 단계 중 앞의 것인 스텝 1에서 수행하는 범주마다에 대한 대표 단어들 및 그들의 가중치를 학습하는 기법을 살펴본다. 대표 벡터를 구하기 위해 먼저 가중치가 0이 아닌 단어들 즉 대표 단어들 및 그들의 가중치를 구해야 한다.

범주 c에 대한 씨앗 단어들의 집합인 S_c는 이 범주를 대표하는 단어들을 처음에 마련하는데 사용하게 된다. 만약, 대표 단어로써 S_c에 속한 단어들만을 사용하게 된다면 이 단어들이 나타나지 않은 많은 문서들에 대해서는 정확한 판단을 하기 어렵기 때문에 시스템 성능의 저하로 연결되게 된다. 그러므로 이러한 S_c 이외에 더 많은 단어들을 대표단어로 학습해야 하며 이 단어들의 가중치 또한 결정하여야 한다.

더 많은 대표 단어들을 학습하기 위해 평균 상호정보(average mutual information)를 이용했다. Y_c를 범주 c에 대한 대표 단어들의 집합이라고 하고 처음에 $Y_c = S_c$라고 설정한 후 부트스트래핑 과정에서 더 많은 대표 단어들을 학습하여 Y_c에 추가 한다. Y_c 내의 단어 y와 Y_c에 아직 들어 있지 않은 단어 x간의 상호 정보는 다음과 같은 식으로 계산한다.

$$M(x,y) = \log \frac{p(y,x)}{p(y)p(x)} \tag{7}$$

이때, $p(y,x) = f(y,x)/N$, $p(x) = f(x)/N$, $p(y) = f(y)/N$이며, $f(y,x)$는 y와 x가 동일 문서에서 함께 나타난 빈도 수(즉 그러한 문서의 수)를 뜻한다. 여기에서 N은 전체 문서집단의 문서 수가 된다(즉 본 논문의 원시말뭉치 T_u의 크기). 단, 추가적으로 학습할 대표 단어들의 가중치를 0과 1사이의 값으로 정규화하기 위해 상호정보 값을 조정할 필요가 있는데 이때 조정한 상호정보 값을 $M'(x,y)$이라고 한다. 조정된 상호 정보값은 1과 0 사이의 실수가 된다. 즉 조정 상호 정보도 역시 상호정보를 정규화 한 것이다.

$$M'(x,y) = \begin{cases} 1, & if\ M(x,y) > Max \\ 0, & if\ M(x,y) < Min \\ \dfrac{M(x,y) - Min}{Max - Min}, & otherwise \end{cases} \tag{8}$$

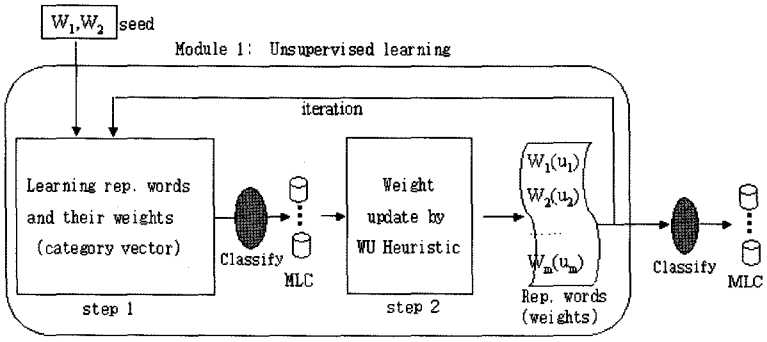

그림 3 비지도 학습 과정

Y_c 안의 y는 범주 c에 대해 이미 학습을 한 대표 단어라 하자. 따라서 가중치 $u_{c,y}$는 이미 정해진 상황이다. 이때 새로운 대표 단어를 학습하는 방법은 다음과 같다. Y_c에 포함되지 않은 각 단어 x에 대해서 우리는 다음과 같은 값 $v(c,y,x)$를 구한다.

$$v(c,y,x) = u_{c,y} M'(x,y) \tag{9}$$

$v(c,y,x)$는 이미 범주 c의 대표 단어인 y를 통하여 새로운 단어 x가 범주 c에 대해 가지는 가중치(또는 중요도)이다. 새로운 단어 x가 이미 범주 c의 대표 단어가 된 단어들과 밀접한 관계를 가지고 있다면 x도 이 범주와 관련이 깊을 가능성이 많다. 즉 x와 y와의 관계를 고려할 때 y가 더욱 중요할수록(즉 y의 가중치가 클수록), 그리고 x와 y사이의 관련도가 클수록 x의 범주 c에 대한 관련도도 커질 것이다. 위의 식 (9)는 이러한 생각을 반영한 식이다. 여기서 y의 중요도는 $u_{c,y}$가 나타내고 x와 y 사이의 관련도는 $M'(x,y)$가 나타내는 것이다. $M'(x,y)$는 $u_{c,y}$의 얼마 만큼이 x의 중요도로 전달되는지를 정한다. 만약 y와 x의 조정된 상호 정보 값 $M'(x,y)$이 1.0이라면 $u_{c,y}$의 전체가 $v(c,y,x)$로 전달된다. 이런 중요도의 전파 작업은 그림 4와 같이 이미 대표단어가 된 모든 단어에 대하여 수행한다.

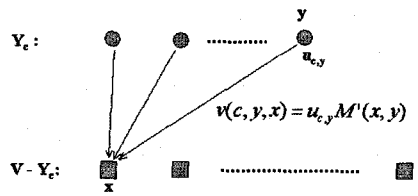

그림 4 중요도의 전파

그렇다면 새로운 단어 x의 범주 c에 대한 중요도 즉 가중치는 다음 식에서처럼 Y_c 안의 모든 y에 대하여 $v(c,y,x)$ 값의 평균을 구함으로써 결정된다.

$$u_{c,x} = \frac{\sum_{y \in Y_c} v(c,y,x)}{|Y_c| \times \beta} \tag{10}$$

β 값은 y로부터 물려 받는 가중치의 정도를 조절하기 위한 상수로서 현재는 값을 1로 하였다. 한 부트스트래핑 단계에서의 하나의 새로운 대표단어의 학습은 다음과 같다: 모든 학습되지 않은 단어들 x에 대한 $u_{c,x}$를 구한 다음, 그 중에서 가장 값이 큰 $u_{c,x}$를 가진 단어 x를 선택하여 이를 Y_c에 추가함으로써 이 x를 대표 단어로 학습한다.

위와 같은 한 대표 단어를 학습하는 부트스트래핑 단계를 계속 반복하여 여러 대표 단어를 수집하게 된다. 이 과정은 해당 범주에 대하여 학습한 단어들의 수가 미리 조절한 어느 임계치에 도달할 때까지 반복되게 된다. 한 대표단어를 학습하면 다음 단계에서는 Y_c의 내용이 변했기 때문에 Y_c에 없는 모든 단어 x에 대하여 다시 $u_{c,x}$ 값을 계산하여야 한다. 각 범주마다 학습할 대표 단어들의 수에 대한 임계치 θ를 얼마로 하여야 할지는 어려운 문제이다. 현재로서 400인 경우에 가장 좋은 성능을 얻을 수 있음을 실험을 통해 알 수 있었다.

이렇게 하여 학습한 대표 단어들에 대한 가중치 $u_{c,x}$를 대표 벡터의 해당 원소로 이용하고, 대표단어가 되지 못한 단어들에 대한 원소는 0으로 한다. 모든 범주에 대하여 대표 벡터를 구하면 시스템은 이를 이용하여 문서를 분류할 수 있다. 그 결과가 그림 3의 두 스텝 사이에 있는 말뭉치 MLC이다. 이 말뭉치는 스텝 2에서의 가중치 재계산에 이용된다.

3.2 대표 벡터의 가중치의 재계산

우리는 앞 3.1 절 스텝 1에서 각 범주에 대한 대표 단어들의 비지도 학습 기법을 소개 했다. 그러나 아직 대표 단어들에 대한 최적의 가중치를 찾은 것은 아니다. 스텝 1의 학습을 수행한 결과를 살펴 보면 매우 일반적인 단어들이 많이 학습 될 수 되며 그들의 u 값 또한 매우 높게 결정된 것을 관찰할 수 있었다. 예를 들면 다음 표에서 "alt.atheism" 범주에 대해 매우 일반적인 단어인 "invalid", "evidence"가 학습되었다. 그러나 "evidence", "invalid"와 같은 일반적인 단어들은 범주 "alt.atheism"을 잘 대표한다고 볼 수 없다. 그러므로 그들의 u 값은 0에 가까운 매우 낮은 값으로 하거나 아예 이런 단어들은 학습되지 않도록 하여야 한다.

이러한 이유로 우리는 스텝 1에서 학습한 단어들의 가중치를 다시 조정하여야 할 필요성을 발견하였다. 이를 위하여 스텝 2에서 가중치를 갱신하도록 한다. 이 기법의 핵심은 정보검색에서 많이 사용하는 tf-idf 가중치 부여 방법에서 사용한 아이디어를 도입하는 것이다. 앞에서도 언급하였듯이 정보검색과 문서분류는 매우 유사한 점이 있다. 따라서 정보검색의 tf-idf 가중치 부여 방법을 문서분류에 적용할 수 있을 것이라는 생각에서 출발하였다. 이를 실현하기 위하여 우리는 표 1에서와 같은 대응 관계를 파악하였다.

표 1 정보검색과 문서분류에서의 가중치 계산 컴포넌트 대응

	정보검색	문서분류
Tf 컴포넌트	$tf_{j,i}$	$\eta_{c,i}$
Df 컴포넌트	df_i	cf_i
가중치	$a_{j,i} \propto \frac{tf_{j,i}}{df_i}$	$u_{c,i} \propto \frac{\eta_{c,i}}{cf_i}$

이렇게 tf-idf를 이용하여 문서에 대한 단어의 가중치를 구하는 기법은 정보검색 시스템에서 성공적으로 이용되어 왔다. 이러한 점에 착안하여 우리는 하나의 범주에 속한 특정한 단어의 문서 출현빈도의 개념을 사용하기로 한다. $e_{c,i}$는 범주 c로 분류된 문서들 중 단어 w_i를 포함한 문서의 수를 나타낸다고 하자. 그렇다면 단어 w_i를 포함한 문서들 중에서 범주 c로 분류되고 동시에 단어 w_i를 포함한 문서들의 비율은 다음과 같이 계산한다.

$$\eta_{c,i} = \frac{e_{c,i}}{df_i} \tag{11}$$

이 때, df_i는 전체 말뭉치 집합인 T_U 중에서 단어 w_i가 출현한 문서의 수를 나타낸다. $n_{c,i}$는 단어 w_i가 범주 c에 나타난 횟수를 나타내기 위한 것인데 여기서는 실제의 횟수($e_{c,i}$)가 아닌 정규화된 횟수를 이용하고 있다. 정규화는 df_i로 나눔으로써 달성한다. 이렇게 정규화된 횟수를 이용하는 이유는 발생 빈도가 높은 단어와 낮은 단어 사이의 형평성을 맞추기 위한 것이다. 즉 발생 빈도가 높은 단어(즉 df_i가 큰 단어)는 $e_{c,i}$도 클 것이다. 따라서 $e_{c,i}$ 만을 tf 카운트로 하면 항상 빈도가 낮은 단어가 불리할 수밖에 없다. 예를 들면 "Pucket" 라는 단어는 전체 말뭉치로 볼 때 발생 빈도가 낮다. 그렇지만 이 단어가 발생하는 대부분의 경우는 baseball 범주의 문서에 나타난다. 따라서 baseball 범주(범주 번호 = c)로 볼 때는 다른 단어와 비교할 때 상대적으로 많이 나타나는 단어로 간주되어야 한다. 그러나 이 범주 c에 실제로 나타난 문서의 수 $e_{c,i}$를 tf로 이용하면 다른 고빈도 단어에 비하여 높게 나타났다고 할 수 없다. 그러나 정규화된 빈도를 사용하기로 한다면 즉 $n_{c,i}$를 이용하면 이런 문제가 해결된다.

문서 분류에 대한 df 컴포넌트는 cf_i이다. 이것은 단어 w_i를 대표단어로 배운 범주의 수이다. 결국 문서분류에서의 가중치는 표 1의 맨 아래 줄에서처럼 $n_{c,i}/cf_i$가 된다. 결국 범주 c에 대한 단어 w_i의 가중치 $u_{c,i}$는 다음 휴리스틱과 같이 주어진다.

• 가중치-갱신 휴리스틱 :

$$u_{c,i} = \frac{\eta_{c,i}}{cf_i} \tag{12}$$

식 (12)에서 cf_i로 나누는 것에 대한 당위성은 cf_i가 클수록 w_i가 많은 범주를 대표하게 되어 모호성이 커지므로 이 경우 이 단어의 중요도를 낮추어 주어야 할 필요성으로 설명될 수도 있다.

범주 c에 대하여 단어 w_i의 가중치를 계산하는 식 (12)을 이용하기 위해서는 범주 레이블이 붙어 있는 문서 집단이 없으면 불가능하다. 우리는 사람이 범주 레이블을 붙인 문서들이 없으므로 이를 극복하기 위해 (3.1절에서 개발된 분류기를 이용하여) 기계가 문서에 대하여 붙인 범주 레이블을 사용한다. 이렇게 분류된 문서들을 기반으로 "가중치-갱신 휴리스틱"을 이용하여 모든 대표 단어들에 대한 새로운 가중치를 구하여 보다 향상된 대표 벡터를 만든다.

3.3 가중치 계산 작업의 반복

위 3.1과 3.2절에서 설명한 스텝 1과 2의 과정을 반복함으로써 학습의 결과를 좀 더 향상시킬 수 있다. 한번의 반복 과정을 한 에폭(epoch)이라고 부르기로 하자. 만약 이전의 에폭에서 나온 결과를 이용하여 다음 에폭의 입력으로 넣어준다면 이전 에폭의 결과보다 좀 더 나은 결과가 나오게 된다. 우리가 취하는 방법론에서 높은 성능을 보장하기 위해서는 각 범주에 대하여 실제 대표가 될 만한 단어들 만을 학습하여야 하며 및 그에 대한 정확한 가중치를 학습하는 것이다.

그러나 스텝 1의 비지도 학습 과정은 너무 일반적이거나 해당 범주를 대표하지 못하는 단어들을 학습하기도 한다는 점이 문제이다. 이 문제를 해결하기 위해서 우리는 다음과 같은 생각을 이용한다. 즉 어떤 범주 c에 대하여 스텝 1에서 너무 일반적인 단어 x를 대표단어로 학습하였다 하자. (실제로 이 단어는 대표 단어로 학습되어서는 안될 단어라고 하자.) 스텝 2에서는 이 단어의 가중치를 매우 낮은 값으로 갱신하게 된다. 그러나 스텝 2는 단어를 대표 단어로부터 제거하는 기능은 없다. 즉 스텝 2는 대표 단어의 제거 기능은 없고 단지 가중치를 보다 정확한 것으로 갱신하여 주는 기능은 있다.

그렇다면 단어 x에 대하여 스텝 2에서 매우 낮은 가중치를 받은 사실을 이용할 수는 없을까? 만약 스텝 1을 다시 한번 더 수행하도록 하고 그 과정에서 x가 스텝 2에서 매우 나쁜 가중치를 받은 사실을 고려하여 가능하면 다른 단어 보다 먼저 대표 단어로 학습되지 못하도록 한다면 x가 학습될 가능성을 낮출 수 있을 것이다. h 에폭의 스텝 1(즉 3.1절에서 설명한 작업 단계)에서 특정 범주에 대하여 단어 x가 학습된다고 가정하자. 만약 x가 이 범주에 대해 좋지 않은 단어라면 같은 에폭의 스텝 2 단계(3.2절에서 설명한 작업 단계)에서 그 단어의 가중치는 낮아질 것이다. 만약 h+1 에폭의 스텝 1 단계에서 낮은 가중치를 갖게 된다면(즉 u 값이 작다면) 단어 x는 이 범주의 대표 단어들 중에서 늦은 부트스트래핑 단계에서 학습되거나 배우는 순위가 너무 뒤로 밀려 결국 임계치 이전에 배우지 못할 수도 있다. 이런 아이디어를 실현하기 위하여 우리는 다음 기법을 사용한다.

한 에폭 h-1에서 어느 단어에 대한 가중치를 갱신했다면, 그 다음 에폭 h의 스텝 1 단계에서는 중요도를 다음 식과 같이 정하도록 한다.

$$\hat{u}^{h}_{c,x} = \frac{\sum_{y \in Y_c} v(c,y,x)}{|Y_c| \times \beta} \times \frac{\eta^{h-1}_{c,x}}{cf^{h-1}_{x}} \quad (13)$$

이 때, n, cf, u의 위 첨자 h는 에폭 번호를 나타낸다. 하지만 에폭 0은 없기 때문에 $\eta^{0}_{c,x}$와 cf^{0}_{x}는 1.0을 취한다. 모듈 1은 위 식 (11)에 대해 h를 증가시키면서 반복적으로 적용하여 보다 정확한 가중치를 구하고 자 한다. 이 때 연속되는 두 에폭의 문서 분류 결과를 비교하여 변화가 작다면 반복은 끝난다. 실험 결과 3번의 반복을 거치면 좋은 성능을 내는 시스템을 얻을 수 있음이 관찰되었다.

이렇게 하여 최종적으로 학습된 대표 벡터를 사용하여 원시말뭉치 T_U의 문서들을 분류하여 범주 레이블을 붙인다. 그 결과로 기계 부착 말뭉치인 T_L을 얻는다.

4. SVM에 의한 최종적인 문서분류기의 학습

최종적인 문서분류기를 얻기 위해서 모듈 2는 모듈 1에서 기계가 레이블을 붙인 말뭉치 T_L을 학습 말뭉치로 사용하여 지도학습을 하게 된다. 본 논문에서는 지도학습 알고리즘으로 현재 문서 분류 시스템에서 가장 높은 성능을 보이는 SVM을 선택하였고, 다중 분류가 가능한 Libsvm-2.81버전[1]을 사용하였으며, 이때 RBF 커널(kernel)을 사용하였다. 여기에서 γ 값은 디폴트(default) 값을 그냥 이용하였고, C 값은 0~1000 사이에서 5씩 증가시키면서 최적의 값을 찾았다. 그 결과 C = 10일 때 최고의 성능을 얻을 수 있었다. 입력 데이타는 앞서 말한 약 12,222개의 단어를 정보 검색 시스템에서 주로 사용되는 식 (14)과 같은 적절한 용어 가중치 계산 방법을 이용하여 가중치를 조절하여 입력하였다.

$$\frac{tf}{MAX(tf)} \times \log\frac{N}{df} \quad (14)$$

모듈 2의 결과로 시스템은 문서 분류기를 얻게 된다. 일반적으로 모듈 2의 지도학습에 의하여 모듈 1에 비하여 더 좋은 성능의 문서분류 시스템을 구할 수 있다고 알려져 있다.

5. 실 험

5.1 데이타 집합

실험에는 Ken Lang이 수집한 20 newsgroups 데이타셋을 사용하였다[11]. 이 데이타셋은 전자우편 문서들을 모은 것으로서 문서분류의 난이도가 상당히 높기 때문이다. 특히 20개의 범주 전체에 대한 분류 문제는 매우 까다로운 것으로 알려져 있다. 왜냐하면 일부 범주는 다른 범주와 의미적으로 겹치기 때문에 범주 결정이 매우 어렵다. 또한 문서 내의 텍스트는 불규칙성이 많이 존재하여 텍스트 처리 자체도 간단하지 않다. 이 데이타셋 이외에도 Reuters 데이타셋과 같이 많이 이용되는 것도 있으나 이것은 대부분의 문서분류기 들에 의한 실험 결과 그 난이도가 높지 않아 20 newsgroups 데이타셋 보다 높은 성능을 보임이 알려져 있다.

따라서 우리는 20 Newsgroups 데이타셋 만에 의한 실험으로 우리 시스템의 성능을 측정하도록 하였다. 특히 우리는 20 newgroups 데이타셋 중에서도 bydate version을 사용하였다[6]. 이는 총 18,846개의 문서로 이루어져 있고, 미리 학습 데이타로 60%, 테스트 데이타 40%로 나누어져 있다. 이것은 Gliozzo[6]의 연구에서와 똑 같은 데이타셋을 이용하여 실험을 함으로써 그들의 결과와 비교를 하기 위한 것이다. Gliozzo의 연구는 최근의 가장 진보된 연구로 간주되므로 이 연구와의 비교를 통해 우리 시스템을 평가할 수 있다고 보기 때문이다. 그들과 마찬가지로 우리는 교차 검증(cross validation)은 하지 않았다.

5.2 실험 결과

• 11-category data 실험

사람이 문서 레이블을 수작업으로 붙인 문서 집단을 이용하여 지도학습 알고리즘을 통하여 구축한 문서 분류 시스템이 최고의 성능을 보이는 것으로 알려져 있다. 이러한 최적의 시스템과 비교하여 우리의 방법에 의해 개발된 시스템이 어느 정도 성능을 보이는 지 알아 보는 것이 본 실험의 목적이다.

이를 위해 우리는 20개의 범주 중 각 범주간의 중복되는 정도가 낮은 11개의 범주를 선택하였다.[2] 이 때 공정한 성능의 비교를 위해 동일한 훈련 말뭉치와 테스트 말뭉치를 사용하였고 같은 자질집합(feature)을 사용하였다.

표 2에서 확인할 수 있듯이 모듈 2의 최종 결과 즉, 모듈 1에서 기계가 레이블을 붙인 말뭉치로 학습한 SVM분류기와 사람이 제공한 정답 레이블로 학습한 SVM(pure-SVM)은 성능의 차이가 거의 없음을 확인

표 2 11-category 데이타에 대한 성능 비교

	F-measure
Ours(스텝 1)	79.22
Ours(스텝 2)	88.22
Ours(SVM)	90.22
Pure-SVM	91.04

1) http://www.csie.ntu.edu.tw/~cjlin/libsvm/

2) 선택된 11개의 범주: alt.atheism, comp.windows.x, misc.forsale, rec.autos, rec.motocycles, rec.sport.baseball, rec.sport.hockey, sci.crypt, sci.med, sci.space, talk.politics.mideast.

할 수 있었다.

• 20-category data 실험

Glizzo[6]는 20-newsgroups 데이타셋의 20개의 범주 모두를 사용한 실험을 하였고, 실험 결과는 표 3과 같다. 이들도 우리와 유사하게 두 단계의 학습을 하는데, 그들이 사용한 LSI(latent semantic indexing)와 GM (Gaussian mixture)이론은 이해하기 쉽지 않고 복잡한 반면, 우리는 단어간의 상호 정보량과 문서 출현빈도등과 같이 단순한 개념을 사용함에도 더욱 향상된 결과를 나타내는 것을 확인 할 수 있었다.

표 3 20개의 범주에 대한 성능 비교

	Ours		Gliozzo et. al.[6]	
20-category Newsgroup dataset (bydate version)	스텝 1	60.71%	LSI	50%
	스텝 2	63.14%	GM	60%
	SVM	71.24%	SVM	65%

본 실험에서는 Gliozzo[6]과의 성능 비교를 위해서 그들이 사용한 실험 데이타를 그대로 이용하였다. 그들은 전체 데이타를 훈련과 실험 부위로 나누는 위치를 변화시키면서 실험하여 그 평균을 구하는 cross validation 실험은 수행하지 않았다. 우리도 직접적인 성능비교를 위해 그들의 실험 데이타 구성을 그대로 이용하였다.

20개의 범주 각각의 F-measure값은 표 4와 같다. 여기에서 알 수 있듯이 일부 범주의 경우 그 성능이 매우 낮음을 알 수 있다. 그 이유는 범주 간에 의미적인 겹침 현상이 있기 때문이다. 앞으로 이 문제를 해결하는 것이 시스템의 성능 향상에 중요함을 알 수 있다.

• 비지도 학습의 반복 효과

모듈 1의 두 단계를 반복하여 수행함으로써 많은 성능의 향상을 얻을 수 있음을 실험을 통하여 관찰하였다 (표 5 참조). 이것은 스텝 2의 결과를 다른 에폭의 스텝 1에게 피드백하여 줌으로써 새로운 단어를 학습할 때 좋지 않은 것들은 될수록 늦게 학습하거나 아예 학습되지 않게 하는 효과를 거둘 것이라는 우리의 추측을 뒷받침하는 결과이다. 이 표에서 각 열은 해당 갯수만큼의 반복만 수행하여 시스템을 구축할 경우에 얻는 성능을 말한다. 보통 3번의 반복을 통해 가장 높은 성능에 도달하여 그 이후에는 크게 변하지 않는 것을 관찰하였다.

Table 5 비지도 학습의 반복을 통한 정보의 피드백 실험

반복 수	1	2	3	4
단계 2(마지막 에폭)	55.06%	61.0%	63.14%	61.29%
SVM	61.82%	70.31%	71.24%	71.18%

• 학습단어 수에 대한 실험

범주마다 몇 개의 단어를 학습하여야 할 지를 결정하기 위해서 우리는 여러 가지 임계치 θ를 변화시켜 가면서 성능을 측정하는 실험을 하였다. 실험 결과 400 개의 단어를 학습하는 것이 가장 성능이 좋은 것으로 관찰되었다. 하지만 200 개를 넘으면 거의 최상의 성능에 매우 근접하는 결과를 얻을 수 있었다. 너무 많은 단어를 학습하는 경우 오히려 성능이 감소하는 것을 확인하였다.

표 6 학습 단어 수 임계치 θ의 효과

단어수 θ	10	100	200	300	400	500
F-measure(%)	66.06	70.45	71.02	71.16	71.24	70.05

6. 결 론

본 논문에서는 문서분류기의 개발을 위하여 원시말뭉치와 씨앗 정보만을 이용하는 준-비지도 학습 기법을 제안하였다. 각 범주의 범주명을 구성하는 한 두 단어를 씨앗 단어로 이용하여 해당 범주를 대표하는 단어들을 부트스트래핑 기법으로 학습하고 그 단어들의 가중치를 조정하여 범주를 대표하는 벡터들을 생성하였다. 이 때 가중치를 업데이트 시키는 방법으로 정보 검색 시스템에서 많이 사용하는 문서 출현 빈도를 응용하였고, 이 과정을 반복함으로써 성능을 향상 시키도록 하였다. 실험 결과 기존의 다른 연구 결과보다 높은 성능을 나타내는 것을 확인 할 수 있었고, 11범주에 대한 실험을 통

표 4 우리 시스템(SVM)의 범주별 성능

범주	F-measure	범주	F-measure
atheism	27.27	sport-hockey	95.23
comp-graphics	63.23	sci-crypt	90.65
ms-win-misc	75.12	sci-electronics	56.48
ibm-pc-hardware	64.28	sci-med	80.30
comp-sys-mac-hardware	75.58	sci-space	89.59
comp-windows-x	78.73	soc-religion-christian	91.45
misc-sale	75.12	talk-politics-guns	57.14
rec-autos	84.84	talk-politics-mideast	84.84
rec-motocycles	86.43	politics-misc	0.00
sport-baseball	91.93	religion-misc	3.18

하여 정답문서로 훈련한 SVM 문서분류기와 성능 차이가 거의 없음을 확인 할 수 있었다. 그러나 20개의 범주를 모두 사용하였을 경우 범주간의 중복되는 부분이 성능을 저하시킴을 확인 할 수 있었다. 향후 각 범주간에 중복이 존재하거나 100% 정확하지 않은 훈련 데이타에 대해서도 높은 성능을 낼 수 있는 연구가 필요하다.

참 고 문 헌

[1] C. Manning and H. Schutze, 1999. Foundations of Statistical Natural Language Processing. The MIT Press.

[2] T. Joachims, 1998. Text categorization with support vector machines: learning with many relevant features. In Proc. of ECML '98, Pages 137-142.

[3] D. Lewis and W. Gale. 1994. A sequential algorithm for training text classifiers, In Proc. of SIGIR-94.

[4] A. Blum and T. Mitchell. 1998. Combining labeled and unlabeled data with co-training. In Proc. COLT-98.

[5] K.P. Nigam, A. McCallum, S. Thrun, and T. Mitchell. 1998. Learning to classify text from labeled and unlabeled documents. In Proc. of AAAI-98.

[6] A. A. Gliozzo, C. Strapparava, and I. Dagan. 2005. Investigating unsupervised learning for text categorization bootstrapping. In Proc. of HLT-2005, October. Pages 129-136.

[7] Y. Ko and J. Seo. 2004. Learning with unlabeled data for text categorization using bootstrapping and feature projection techniques. In Proc. of the ACL-04, Barcelona, Spain, July.

[8] B. Liu, X. Li, W.S. Lee, and P.S. Yu. 2004. Text classification by labeling words. In Proc. of AAAI-04, San Jose, July.

[9] G. Salton and M. McGill. 1983. Introduction to Modern Information Retrieval. McGraw-Hill.

[10] Y. Yang and J.P. Pederson. 1997. Feature selection in statistical learning of text categorization. In Proc. of ICML '97, Pages 412-420.

[11] A. McCallum and K. Nigam. 1999. Text classification by bootstrapping with keywords, EM and shrinkage. In ACL-99-Workshop on Unsupervised Learning in Natural Language Processing.

[12] S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman. 1990. Indexing by latent semantic analysis. Journal of the American Society of Information Science.

[13] A. Dempster, N. M. Laird and D. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. J. of the Royal Stat. Society, B:39, Pages 1-38.

[14] R. Ghani. 2002. Combining labeled and unlabeled data for multiclass text categorization. In Proc. of ICML-02.

[15] A. Gliozzo, C. Strapparava, and I. Dagan. 2004. Unsupervised and supervised exploitation of semantic domains in lexical disambiguation,. Computer Speech and Language, 18:275-299.

[16] A.K. Jain and R.C. Dubes. 1988. Algorithms for Clustering Data. Engle-wood Cliffs, NJ: Prentice Hall.

[17] T. Joachims, 1999. Estimating the Generalization Performance of an SVM Efficiently. In Proc. of ICML' 2000, Pages 431-438.

[18] Y. Ko and J. Seo. 2000. Automatic text categorization by unsupervised learning. In Proc. of COLING 2000.

[19] A. McCallum and K. Nigam. 1998. A comparison of event models for naive Bayes text classification. In Proc. of AAAI-98 Workshop on Learning for Text Categorization.

[20] N. Slonim, N. Friedman, and N. Tishby, 2002. Unsupervised document classification using sequential information maximization, In Proc. of SIGIR '02, Pages 129-136.

[21] V. Vapnik. 1995. The nature of statistical learning theory.

[22] C. Burges, 1998. A Tutorial on Support Vector Machines for Pattern Recognition. Data Mining and Knowledge Discovery, vol. 2, no. 2,

[23] N., Cristianini J. Shawe-Taylor2000. An introduction to Support Vector and other kernel-based learning methods. Cambridge Univ. Press.

노 대 욱

2004년 8월 연세대학교 컴퓨터정보통신공학부 졸업(학사). 2007년 2월 연세대학교 컴퓨터정보통신공학부 졸업(석사). 2006년 7월~현재 코오롱베니트 직원. 관심분야는 정보검색, 데이타베이스

이 수 용

1992년 경희대학교 수학과 졸업(이학박사). 2004년 연세대학교 컴퓨터과학과 졸업(공학박사). 2004년 9월~현재 연세대학교 컴퓨터정보통신공학부 조교수. 관심연구분야는 퍼지이론 및 응용, 기계학습, 패턴분류, 데이타마이닝

나 동 열

1978년 서울대학교 전자공학과 졸업(학사). 1980년 KAIST 전산학과 졸업(석사). 1989년 미시간주립대 전산학과 졸업(박사). 1991년 3월~현재 연세대학교 컴퓨터정보통신공학부 교수. 관심분야는 자연어처리, 정보검색, 인공지능