정보검색 기술을 이용한 비교사 학습 기반 문서 분류 시스템 개발

노대욱 이수용 나동열
연세대학교 정보통신공학부
{dwnoh2272, sylee, dyra}@yonsei.ac.kr

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

Dae-Wook Noh Sooyong Lee Dong-Yul Ra
Computer & Telecommunication Engineering Div., Yonsei University

요 약

문서분류기의 개발에 있어 교사학습기법을 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 비교사 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 개발할 수 있음을 제시하였다.

1. 서론

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 자동분류 시스템은 그 중요성이 날로 커지고 있다.

이러한 문서 분류기의 개발에는 주로 기계 학습 방법을 사용한다[13]. 기존의 교사 학습(supervised learning) 알고리즘을 이용한 연구들을 살펴보면 문서 분류기의 성능이 매우 높은 것을 확인 할 수 있다[8, 11]. 하지만 이런 높은 성능에 이르기 위해서는 사람이 수작업으로 범주 레이블(label)을 붙인 충분한 양의 학습 말뭉치가 미리 준비되어 있어야 한다. 그러나 사람들이 직접 레이블이 붙은 학습 말뭉치를 준비하기 위해서는 많은 시간과 노력이 필요한 일이므로 쉬운 일이 아니다.

이러한 문제점을 해결하기 위해 최근의 몇몇 논문들이 비교사 학습(unsupervised learning)에 기반하여 원시(raw) 말뭉치에 범주 레이블을 붙이고 이를 이용하여 교사학습 알고리즘을 훈련시켜 문서 분류 시스템을 개발하자는 제안이 있었다 [6, 10, 12]. 본 논문에서는 이러한 패러다임(paradigm)에 기반한 문서 분류기의 새로운 개발 기법을 소개하고자 한다.

아무런 정보도 제공되지 않은 상황에서의 비교사 학습 기법은 거의 불가능하므로, 우리가 따르는 패러다임에서는, 초기 입력 정보로 범주 레이블이 없는 문서(원시 말뭉치)들과 각 범주를 대표하는 씨앗(seed)

단어들을 제공한다[1][6, 10]. 이러한 데이터를 이용하여 먼저 비교사 학습 기법을 통하여 각 범주에 대한 정보를 학습한다. 그리고 이를 이용하여 각 원시 말뭉치의 각 문서를 분류하여 범주 레이블을 부착한다. 이 결과로 기계에 의한 범주 표지 부착 말뭉치(machine-labeled corpus)를 얻는다. 이렇게 얻은 말뭉치를 이용하여 교사 학습 알고리즘을 학습하여 최종의 문서 분류기를 얻게 된다. 즉 주어진 패러다임에서는 첫단계로 비교사 학습 알고리즘을 이용하고 (모듈 1), 다음 단계로 교사 학습 알고리즘을 이용하는 단계(모듈 2) 를 거친다.

본 논문의 연구에서는 정보검색에서 각 문서가 벡터로 표현되는 것처럼 각 범주들을 문서와 같이 벡터로 가능한한 정확하게 나타내고(이를 범주 대표 벡터라 함) 이 대표 벡터들을 사용하여 정보검색에서 처럼 cosine 계수 등과 같은 비교 기법을 이용하여 문서를 분류하는 것이다.

본 논문의 첫째 특징은 비교사 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. 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.

[1] 따라서 엄밀히 말하면 우리의 기법은 준-비교사 (semi-supervised) 학습 기법의 일종이다.

실험을 통하여 본 논문에서 제안한 기법이 다른 연구에서 제안한 기법보다 우수한 성능을 나타냄을 확인할 수 있었다.

우리의 시스템은 학습과정에서 순차적으로 적용되는 두 개의 모듈로 구성되어 있다. **모듈 1에서는 레이블이 없는 원시 문서들을 이용하여 학습을 수행한 후 이들 문서에 범주를 결정하여 붙이게 된다. 첫 모듈은 레이블이 없는 문서 즉 원시 문서들만 사용하게 되므로 비교사 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, 10]. 예를 들어 20 newsgroup data set의 한 범주인 "rec.sport.baseball" 에 대해서는 씨앗 단어로 공통된 범주명인 *rec*와 *sport*를 제외한 *baseball*만을 사용하게 된다.**

모듈 2 에서는 교사 학습 방법으로 문서 분류기를 얻는다. 모듈 2 가 사용하는 학습 문서 집단은 모듈 1 의 결과인 기계가 범주를 붙여준 문서들이 된다. 모듈 2 는 대부분의 기존 연구에서와 같이 문서 분류에서 좋은 성능을 보이는 Support Vector Machine (SVM)을 사용하였다[6].

모듈 2의 결과로 얻는 분류 시스템이 최종적으로 우리가 얻는 문서 분류기이다.

2. 문서에 대한 범주의 결정

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. (집합 V 와 C 는 각각 색인어 집합 및 범주 집합을 나타내는 데 순서화된(ordered) 집합이다.) 예를 들어 c 번째 범주는 대표 벡터 R_c 로 나타낸다.

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

만약 V 의 j 번째 단어가 범주 c 를 대표하는 단어가 되기 위해서는 해당되는 R_c의 $u_{c,j}$가 0 이 아닌 값을 가져야 한다. 값이 0 인 위치에 대응되는 단어들은 이 범주를 대표하는 단어가 아닌 것이다. $u_{c,j}$ 의 값은 w_j 가 얼마만큼 범주 c 를 대표하는 대표성이 있는 지를 나타내는 가중치이다.

각 범주에 대한 대표 벡터가 모두 마련되어 있으면 임의의 문서 D 의 범주를 결정할 수 있다. 이를 위해 문서 D 와 문서 R 의 유사도를 이용하는 것이다. 이는 정보검색에서 효율적으로 사용되는 cosine 값을 이용해 계산한다[17].

$$sim(D, R_c) = \cos ine(D, R_c) \quad (2)$$

그렇다면 이 문서에 대한 범주는 다음과 같이 결정할 수 있다.

$$Category_{chosen} = ARGMAX_c sim(D, R_c) \quad (3)$$

3. 범주 대표 벡터의 계산

3.1. 평균 상호 정보를 이용한 대표 단어 및 그 가중치 결정

대표 벡터를 구하기 위해 먼저 가중치가 0 이 아닌 단어들 즉 대표 단어들 및 그들의 가중치를 구해야 한다.

범주 c 에 대한 씨앗 단어들의 집합인 S_c 는 이 범주를 대표하는 단어들을 처음에 마련하는데 사용하게 된다. 만약, 대표 단어로써 S_c 에 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_c 를 범주 c 에 대한 대표 단어들의 집합이라고 하고 처음에 $Y_c = S_c$ 라고 설정한 후 부트스트래핑 과정에서 더 많은 대표 단어들을 학습하여 Y_c 에 추가 한다. Y_c 내의 단어 y 와 Y_c 에 아직 들어 있지 않은 단어 x 간의 상호 정보는 다음과 같은 식으로 계산한다.

$$M(x,y) = \log \frac{p(y,x)}{p(y)p(x)} \quad (4)$$

이때, $p(y,x) = f(y,x)/N$, $p(x) = f(x)/N$, $p(y) = f(y)/N$ 이며, $f(y,x)$는 y 와 x 가 동일 문서에서 함께 나타난 빈도 수 (즉 그러한 문서의 수)를 뜻한다. 여기에서 N 은 전체 문서집단의 문서 수가 된다 (즉 본 논문의 원시말뭉치 T_u 의 크기). 단, 추가적으로 학습할 대표 단어들의 가중치를 0 과 1 사이의 값으로 정규화하기 위해 상호정보 값을 조정할 필요가 있는데 이때 조정한 상호정보 값을 $M'(x,y)$ 이라고 한다.

$$M'(x,y) = \begin{cases} 1, & if\ M(x,y) > Max \\ 0, & if\ M(x,y) < Min \\ \dfrac{M(x,y) - Min}{Max - Min}, & otherwise \end{cases} \quad (5)$$

Y_c 안의 y 는 범주 c 에 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_c 에 포함되지 않은 각 단어 x 에 대해서 우리는 다음과 같은 값 $v(c,y,x)$를 구한다.

$$v(c,y,x) = u_{c,y} M'(x,y) \quad (6)$$

$v(c,y,x)$는 이미 범주 c 의 대표 단어인 y 를 통하여 새로운 단어 x 가 범주 c 에 대해 가지는 가중치(또는 중요도)이다. $M(x,y)$에 의해 $u_{c,y}$ 의 얼마만큼이 x 의 중요도로 전달되는지가 정해진다. 만약 y 와 x 의 조정된 상호 정보 값 $M(x,y)$이 1.0 이라면 $u_{c,y}$ 의 전체가 $v(c,y,x)$로 전달된다. 그렇다면 새로운 단어 x 의 범주 c 에 대한 중요도 즉 가중치는

다음 식에서처럼 Y_c 안의 모든 y 에 대하여 $v(c,y,x)$ 값의 평균을 구함으로써 결정된다.

$$u_{c,x} = \frac{\sum_{y \in Y_c} v(c,y,x)}{|Y_c| \times \beta} \quad (7)$$

β 값은 y 로부터 물려 받는 가중치의 정도를 조절하기 위한 상수로서 현재는 값을 1 로 하였다.

새로운 단어 x 를 위한 $u_{c,x}$ 계산은 $v(c,y,x)$의 평균값을 사용하는데, 이는 미리 학습을 한 대표 단어들의 집합인 Y_c 에 포함된 모든 y 에 대한 평균값이 된다.

모든 학습되지 않은 단어 들 x 에 대한 $u_{c,x}$ 를 구한 다음 그 중에서 가장 값이 큰 $u_{c,x}$ 를 가진 단어 x 를 선택하여 이를 Y_c 에 추가함으로써 대표 단어로 학습한다.

위와 같은 한 대표단어를 학습하는 작업을 계속 반복하여 여러 대표 단어를 수집하게 된다. 이 과정은 해당 범주에 대하여 학습한 단어들의 수가 미리 조절한 어느 임계치에 도달할 때까지 반복되게 된다. Y_c 의 내용이 변했기 때문에 Y_c 에 없는 모든 단어 x 에 대하여 이전 단계에서 계산한 $u_{c,x}$ 값을 다시 계산하여야 한다.

이렇게 하여 구한 대표 단어들에 대한 가중치를 대표 벡터의 해당 원소로 이용하고, 대표단어가 되지 못한 단어들에 대한 원소는 0 으로 한다. 모든 범주에 대하여 대표 벡터를 구하면 시스템은 이를 이용하여 문서를 분류할 수 있다 (이렇게 하여 생성된 시스템을 A1.1 이라 부름).

3.2. 대표 벡터의 가중치의 재계산

우리는 앞 절 3.1 에서 각 범주에 대한 대표 단어들의 학습 방법을 소개 했다. 각 범주마다 학습할 대표 단어들의 수 θ를 얼마로 하여야 할 지는 어려운 문제이다. 현재는 θ = 200 인 경우에 대한 한 가지만을 실험하였다. θ 값을 여러 가지로 변경하면서 어느 경우에 가장 좋은 결과를 얻는 지를 알아 보는 실험이 필요하다.

앞 절 3.1 의 학습의 결과를 이용함으로써 각 범주별로 식 (1)의 대표 벡터를 준비한다. 만약 $y \in Y_c$ 가 V 집합의 j 번째 단어라면 R_c 의 j 번째 원소로 $u_{c,y}$ 를 넣는다. 표시할 수 있다. 주어진 범주 c 에 대하여 대표 단어로 학습이 되지 못한 단어들에 대해서는 R_c 안에 해당 원소의 값으로 0 을 넣는다.

이제 식 (3)를 사용하여 원시말뭉치 T_U 안의 문서들을 분류할 준비가 되었다. 그러나 아직 대표 단어들에 대한 최적의 가중치를 찾은 것은 아니다. 하나의 이유는 [6]에서 제안된 것처럼 모든 범주에 대해 정규화 방식을 다르게 하기 때문이다.

3.1 절의 기법을 이용할 경우 매우 일반적인 단어들이 많이 학습 될 수 있으며 그들의 u 값 또한 매우 높게 측정된다. 예를 들면 다음 표에서 "atheism" 범주에 대해 매우 일반적인 단어인 "invalid"가 학습되었다.

Category	Words learned	
atheism	evidence	invalid

그러나 "evidence", "invalid"와 같은 일반적인 단어들은 범주 "atheism"을 잘 대표한다고 볼 수 없다. 그러므로 그들의 u 값은 제거되어야만 한다. 이러한 이유로 우리는 단어들의 가중치를 결정하는데 보다 효과적인 특수한 기법을 소개한다. 이 기법을 이용하여 앞에서 학습한 대표단어들에 대하여 가중치를 갱신한다.

다음 식과 같이 문서 d_i 에서 나타난 단어 w_j 의 가중치 $u_{j,i}$ 는 정보검색 분야에서 널리 이용되는 tf 와 df 값을 사용한다. 이 때, df_i 는 w_i가 나타난 문서들의 수이다.

$$u_{j,i} \propto \frac{tf_{j,i}}{df_i} \quad (8)$$

이렇게 tf-idf 를 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 c 로 분류된 문서들 중 단어 w_i 를 포함한 문서의 수를 나타낸다고 하자. 그렇다면 단어 w_i 를 포함한 문서들 중에서 범주 c 로 분류되고 동시에 단어 w_i 를 포함한 문서들의 비율은 다음과 같이 계산한다.

$$\eta_{c,i} = \frac{e_{c,i}}{df_i} \qquad (9)$$

이 때, df_i는 전체 말뭉치 집합인 T_U 중에서 단어 w_i 가 출현한 문서의 수를 나타낸다. 우리는 더 큰 $\eta_{c,i}$ 값일수록 단어 w_i 는 범주 c 를 더욱 잘 대표한다고 볼 수 있다. 만약 $e_{c,i}$ 가 높다면 (범주 c 로 분류된 w_i 를 가진 문서가 많다면), 심지어 df_i 가 높더라도 (w_i 를 포함한 문서 수가 많더라도) w_i 는 c 에게 매우 중요한 단어라 할 수 있다.

식 (8) 에서 $tf_{j,i}$ (문서 j 에 나타난 단어 w_i 의 수)가 크면 문서 j 에 대한 단어 i 의 가중치가 높아지듯이, 식 (9)이 말하는 바는 범주 c 에 단어 w_i 가 많아 나타난다면 (즉 $e_{c,i}$ 가 높다면) w_i 는 범주 c 에 중요한 단어로 볼 수 있다. 분모로 사용된 df (document frequency) 는 이 값들의 정규화를 위한 것이다.

결국 범주 c 에 대한 단어 w_i 의 가중치 $u_{c,i}$ 는 다음 휴리스틱과 같이 주어진다.

• 가중치-갱신 휴리스틱 :

$$u_{c,i} = \frac{\eta_{c,i}}{cf_i} \qquad (10)$$

이 때, cf_i 는 단어 w_i 를 대표 단어로 가지고 있는 범주의 수이다. 식 (10)에서 cf_i 로 나눈 이유는 cf_i 가 클수록 w_i 가 많은 범주를 대표하게 되어 모호성이 커지므로 이 경우 이 단어의 중요도를 낮추어 주기 위한 것이다.

범주 c 에 대하여 단어 w_i 의 가중치를 계산하는 식 (10)을 이용하기 위해서는 범주 레이블이 붙어 있는 문서 집단이 없으면 불가능하다. 우리는 사람이 범주 레이블을 붙인 문서들이 없으므로 이를 극복하기 위해 (3.1 절에서 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. (이렇게 하여 나온 분류 시스템을 A1.3 이라 부름.)

3.3. 가중치 계산 작업의 반복

위 3.1 과 3.2 절의 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 되는 단어 및 그에 대한 정확한 가중치를 학습하는 것이다. 그러나 때때로 시스템은 너무 일반적이거나 해당 범주를 대표하지 못하는 단어들을 학습하기도 한다.

h 에폭의 A1.1 단계(3.1 절에서 설명한 작업 단계)에서 특정 범주에 대하여 단어 x 가 학습된다고 가정하자. 만약 x 가 이 범주에 대해 좋지 않은 단어라면 같은 에폭의 A1.3 단계 (3.2 절에서 설명한 작업 단계)에서 그 단어의 가중치는 낮아질 것이다. 만약 h+1 에폭의 A1.1 단계에서 낮은 가중치를 갖고

있다면 단어 x 는 이 범주의 대표 단어들 중에서 다른 단어들보다 순위가 낮아질 것이다.

한 에폭 $h-1$ 에서 어느 단어에 대한 가중치를 갱신했다면 그 다음 에폭 h 의 A1.1 단계에서는 다음과 같이 학습을 하도록 한다.

$$\hat{u}_{c,x}^{h} = \frac{\sum_{y \in Y_c} v(c,y,x)}{|Y_c| \times \beta} \times \frac{\eta_{c,x}^{h-1}}{cf_x^{h-1}} \quad (11)$$

이 때, η, cf, u 의 위 첨자 h 는 에폭 번호를 나타낸다. 하지만 에폭 0 은 없기 때문에 $\eta_{c,x}^{0}$ 와 cf_x^{0} 는 1.0 을 취한다. 모듈 1 은 위 식 (11)에 대해 h 를 증가시키면서 반복적으로 적용하여 보다 정확한 가중치를 구하고 자 한다. 이 때 연속되는 두 에폭의 문서 분류 결과를 비교하여 변화가 작다면 반복은 끝난다. 실험 결과 3 번의 반복을 거치면 최고의 가중치 값을 얻는 것이 관찰되었다.

이렇게 하여 최종적으로 학습된 대표 벡터를 사용하여 원시말뭉치 T_U 의 문서들을 분류하여 범주 레이블을 붙인다. 그 결과로 기계 부착 말뭉치인 T_L 을 얻는다.

4. SVM 에 의한 최종적인 문서분류기의 학습

최종적인 문서분류기를 얻기 위해서 모듈 2 는 모듈 1 에서 기계가 레이블을 붙인 말뭉치 T_L 을 학습 말뭉치로 사용하여 교사학습을 하게 된다. 본 논문에서는 현재 문서 분류 시스템에서 높은 성능을 보이는 SVM 을 선택하였고, 다중 분류가 가능한 Libsvm-2.81 버전 [2] 을 사용하였으며, 이때 RBF 커널(kernel)을 사용하였다. 여기에서 감마값은 디폴트(default) 값을 그냥 이용하였고, C 값은 0~1000 사이에서 5 씩 증가시키면서 최적의 값을 찾았다. 그 결과 C=10 일 때 최고의 성능을 얻을 수 있었다. 입력 데이터는 앞서 말한 약 12,000 개의 단어를 정보 검색 시스템에서 주로 사용되는 식(12)과 같은 적절한 용어 가중치 계산 방법을 이용하여 가중치를 조절하여 입력하였다.

$$\frac{tf}{MAX(tf)} \times \log\frac{N}{df} \quad (12)$$

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

5. 실험

5.1. 데이터 집합

실험에는 Ken Lang 이 수집한 20 newsgroup data 가 사용되었다. 이 말뭉치는 많이 알려진 Usenet 의 20 개의 news group 을 포함하며, 하나의 그룹은 하나의 범주로 볼 수 있다. 우리는 newgroup data 중 bydate version 을 사용하였다[6]. 이는 총18,846개의 문서로 이루어져 있고, 학습 데이터로 이 중 60%를 사용하였으며 나머지 40%는 테스트를 하기 위해 사용하였다.

이 60%의 학습 문서는 식 (4)의 상호정보를 구하는 데 이용된다. 이를 바탕으로 하여 이하의 수식들의 값을 계산함으로써 훈련이 수행된다.

5.2. 실험 결과

- 11-category data 실험

사람이 문서 레이블을 수작업으로 붙인 문서 집단을 이용하여 교사학습 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.

[2] http://www.csie.ntu.edu.tw/~cjlin/libsvm/

이를 위해 우리는 20 개의 범주 중 각 범주간의 중복되는 정도가 낮은 11 개의 범주를 선택하였다[3]. 이때 공정한 성능의 비교를 위해 동일한 훈련 말뭉치와 테스트 말뭉치를 사용하였고 같은 자질집합(feature)을 사용하였다.

[표 1] 11-category 데이타에 대한 성능 비교

	F-measure
Ours(A1.1)	79.22
Ours(A1.3)	88.22
Ours(SVM)	90.22
Pure-SVM	91.04

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

• 20-category data 실험

Glizzo et al. [6]는 newsgroup data 의 20 개의 범주 모두를 사용한 실험을 하였고, 실험 결과는 [표 2]와 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 상호정보량과 문서 출현빈도등과 같이 단순한 개념을 사용함에도 더욱 향상된 결과를 나타내는 것을 확인 할 수 있었다.

[표 2] 20 개의 범주에 대한 성능 비교

	Ours		Gliozzo et. al.[6]	
20-category Newsgroup data (bydate version)	A1.1	60.71%	LSI	50%
	A1.3	63.14%	GM	60%
	SVM	71.24%	SVM	65%

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

20 개의 범주 각각의 F-measure 값은 다음과 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.

[표 3] 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

6. 결론

본 논문에서는 씨앗 정보만을 이용한 준 비교사 학습 기법을 제안하였다. 각 범주를 구성하고 있는

[3] 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.

범주명만을 이용하여 해당 범주를 대표하는 단어들을 학습하고 그 단어들의 가중치를 부트스트래핑 기법으로 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, 11 범주에 대한 실험을 통하여 정답문서로 훈련한 SVM 문서분류기와 성능 차이가 거의 없음을 확인 할 수 있었다. 그러나 20 개의 범주를 모두 사용하였을 경우 범주간의 중복되는 부분이 성능을 저하시킴을 확인 할 수 있었다. 본 논문에서는 20-news group 데이터 셋에 대해서만 실험하였으나 로이터 데이터와 같은 다른 데이터에 대하여도 실험을 하여야 할 것으로 생각된다. 향후 각 범주간에 중복이 존재하거나 100% 정확하지 않은 훈련 데이터에 대해서도 높은 성능을 낼 수 있는 연구가 필요하다.

참고 문헌

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

[2] 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.

[3] 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.

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

[5] 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.

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

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

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

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

[10] 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.

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

[12] 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.

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

[14] 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.

[15] 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.

[16] 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.

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

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

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

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

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

[22] N., Cristianini J. Shawe-Taylor 2000. An introduction to Support Vector and other kernel-based learning methods. Cambridge Univ. Press.