Naïve 분류기를 이용한 웹 기반 문서 분류기 구현

박제현*, 최광복*, 한주현*, 최원종**, 양재영**, 최중민*
*한양대학교 컴퓨터공학과
**㈜ 오픈베이스
e-mail : jhpark@islab.hanyang.ac.kr

Implementation of Web-based Document Classification System using Naïve Classifier

Jea-Hyun Park*, Kwang-Bok Choi*, Ju-Hyun Han*, Won-Jong Choi**,
Jaeyoung Yang**, Joongmin Choi*
*Dept. of Computer Science & Engineering, Hanyang University
**Openbase Co. Ltd.

요 약

베이지안 확률 모형은 문서 분류에서 널리 사용되는 이론이다. 그러나, 실제로 베이지안 이론에 기초하여 만들어진 시스템은 처리 시간이 많이 소요된다는 단점을 가지고 있다. 이 논문에서는 문서 분류 작업에 있어 기존의 베이지안 모형을 구현함과 동시에 여러 가지 방법을 통해 시간적인 측면을 개선한 시스템을 구현하였다.

1. 서론

확률 모형에 기반을 둔 문서 분류 시스템에서 널리 사용되는 방법으로 베이지안 모형을 들 수 있다. 그리고, 베이지안 모형이 갖는 높은 복잡도와 긴 처리 시간을 완화시키기 위해, 기존의 베이지안 모형에 여러 독립 가정을 부여한 것이 베이지안 Naïve 모형이다. 베이지안 혹은 Naïve 모형을 이용한 문서 분류 시스템을 구현하는 경우, 실제로 사용하기에는 어려운 단점들을 가지고 있다. 이 논문에서는 그런 단점들을 극복할 수 있는 여러 가지 방법을 통해 시간에 관련된 성능을 개선한 문서 분류 시스템을 구현하였다.

2. 베이지안 분류기

문서 d_j 를 임의의 클래스 c_i 로 분류하는 기준은 d_j 와 c_i 가 단어 w_k 의 모음으로 구성되어 있다고 할 때, 다음과 같은 조건부 확률의 형태로 나타난다.

$$P(c_i \mid d_j) \quad \begin{array}{l} c_i = \{w_{i1}, w_{i2}, \cdots, w_{i|T|}\} \\ \vec{d}_j = < w_{j1}, w_{j2}, \cdots, w_{j|T|} > \\ T : \text{모든 클래스에서 사용된 단어의 집합} \end{array} \tag{1}$$

베이지안 법칙을 적용한 뒤, likelihood 확률을 나타내면 다음과 같다.

$$P(\vec{d}_j \mid c_i) = \prod_{k=1}^{|T|} P(w_{jk} \mid c_i, w_{jk'}; k' < k) \tag{2}$$

순서에 따른 인과 관계를 제거하고, 출현 빈도수에 대한 분포를 출현 여부에 대한 분포로 변경하면 다음과 같은 베이지안 Naïve likelihood 확률 모형을 얻게 된다.

$$P(\vec{d}_j \mid c_i) = \prod_{k=1}^{|T|} P(w_{jk} \mid c_i) \tag{3}$$

위 식에서 $P(w_{jk}|c_i)$는 클래스 i가 단어 w_{jk}를 야기했을 확률이 되며 같은 형태의 단어에 관해서 클래스 내부의 단어에 대한 확률로 모델링하면 아래와 같이 정리할 수 있다.

$$P(w_{jk} \mid c_i) = \left(\frac{P(w_{ik}=1 \mid c_i)}{1 - P(w_{ik}=1 \mid c_i)} \right)^{w_{jk}} (1 - P(w_{ik} \mid c_i)) \quad (3\text{-}1)$$

(3-1)의 식을 (3)에 대입한 뒤 로그 함수의 형태로 나타내고, 문서 *j*와 관계없이 항상 같은 값을 갖는 항을 제거하면, 결과적으로 다음의 베이지안 Naïve 확률 모형을 얻을 수 있다. [1]

$$\log_2 P(c_i \mid \vec{d}_j) \cong \sum_{k=1}^{|T|} w_{jk} \log_2 \frac{P(w_{ik}=1 \mid c_i)}{1 - P(w_{ik}=1 \mid c_i)} - \log_2 P(\vec{d}_j) \quad (4)$$

이 시스템에서는 최상위 클래스를 그 문서가 해당하는 클래스로 간주하였다.

$$\arg\max_i \ \log_2 P(c_i \mid \vec{d}_j) \quad (5)$$

3. 특징 선택

기본적인 베이지안 모형에서는 학습 과정을 통해 추출된 단어를 많이 고려하면 많이 고려할수록 성능이 향상되는 반면에, 베이지안 Naïve 모형은 어떤 학습 데이터를 사용했느냐에 따라서 성능이 향상될 수도, 그렇지 않을 수도 있다. 그리고 고려하려는 단어 수가 많아질수록 그만큼 분류에 소요되는 시간이 길어진다.

그래서, 특징 선택이라는 과정을 통해서 1차 학습 후 선택된 특징만을 사용하여 문서와 클래스 사이의 관련도를 측정하게 된다. 관련도를 측정하는 방법은 여러 가지가 존재하며 대표적인 것들은 다음과 같다.

3.1 TF-IDF

TF-IDF(Term Frequency - Inverted Document Frequency)는 전통적인 정보 검색 분야에서 많이 사용되는 방법으로, 문서의 구조나 단어 사이의 인과 관계는 고려하지 않고 각 단어의 출현 유형 및 빈도수에 의거하여 문서에 대한 단어의 기여도를 측정하는 방법이다. TF와 DF는 다음과 같이 정의된다.

tf_{ik} : 클래스 i에서 단어k가 노출된 빈도수
df_k : 단어k가 노출된 클래스 수

한 클래스에 자주 출현하는 단어는 그만큼 그 클래스에 기여하는 바가 크다고 생각할 수 있지만, 반대로 여러 클래스에서 자주 나타나는 경우에는 빈도수에 비해서 클래스에 대한 기여도가 낮다고 할 수 있다. 그래서 TF-IDF는 다음과 같이 측정한다.[2]

$$TFIDF_{ik} = tf_{ik} \times \log \frac{N}{df_k} \qquad N : \text{전체 클래스 수} \quad (6)$$

3.2 정보 이득(Information Gain)

정보 이론에서 엔트로피를 응용하여 어떤 단어가 클래스에 얼마만큼 기여하는가를 측정하기 위해 정보 이득을 사용할 수 있다. 정보 이득은 어떤 단어의 출현 정보를 아는 상태와 그렇지 않은 상태 사이에서 엔트로피의 변화량을 통해 단어의 클래스 기여도를 측정하게 된다. 일반적인 정보 이득은 다음과 같이 나타낸다.[3]

$$IG(A,V) = H(A) - \sum_{v \in V} \frac{|A_v|}{|A|} H(A_v) \quad (7)$$

이를 문서 분류에 적용하기 위해 다음을 사용하였다.

$$\begin{aligned} IG(c_i, w_k) &= 1 - \sum_{w_k \in \{0,1\}} P(w_k) H(c_i, w_k) \\ &= 1 - \{P(w_k) H(c_i, w_k) + P(\overline{w}_k) H(c_i, \overline{w}_k)\} \end{aligned} \quad (8)$$

엔트로피와는 조금 다르게 정보 이득은 정보량의 차이를 이야기하는 것이고, 그 차이가 크면 클수록 단어가 미치는 영향이 크다고 볼 수 있다. w_k가 c_i에 영향을 미치지 않는다면 정보 이득은 0이 된다.

3.3 상호 정보(Mutual Information)

어떤 두 확률 변수가 얼마나 서로 연관을 갖고 있는가를 측정하기 위한 방법으로 정보 이론의 상호 정보가 있다. 상호 정보는 일반적으로 서로 연관이 있다고 생각되는 두 개의 확률 변수에 대하여 다음과 같이 정의된다.[4]

$$MI(A,B) = \sum_B \sum_A P(A \wedge B) \log \frac{P(A \wedge B)}{P(A) \cdot P(B)} \quad (9)$$

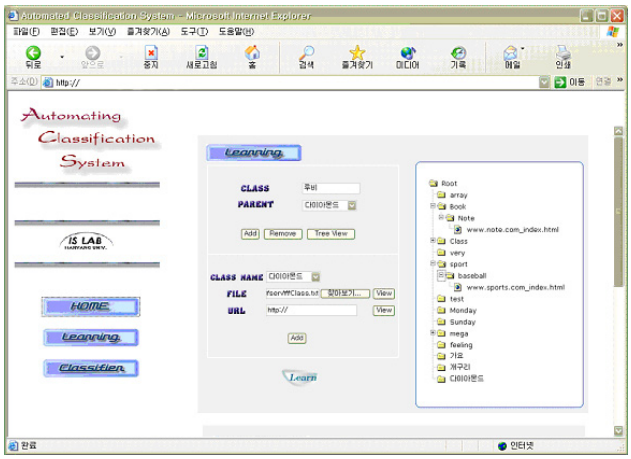
A와 B가 서로 어떤 연관을 갖고 있을 경우, *MI*(*A*, *B*)값은 커지며 반대의 경우 0에 가깝게 된다. 특별히 확률 변수 A와 B가 독립인 경우 *MI*(*A*, *B*)값은 0이 된다. 이 방법을 특징 선택에 적용하면 다음과 같다.[5]

$$\begin{aligned} MI(c_i, w_k) &= \sum_{c_i \in \{0,1\}} \sum_{w_k \in \{0,1\}} P(c_i \wedge w_k) \log \frac{P(c_i \wedge w_k)}{P(c_i) \cdot P(w_k)} \\ &= \sum_{c_i \in \{0,1\}} \sum_{w_k \in \{0,1\}} P(w_k \mid c_i) \cdot P(c_i) \log \frac{P(w_k \mid c_i) \cdot P(c_i)}{P(c_i) \cdot P(w_k)} \\ &= P(w_k \mid c_i) \cdot P(c_i) \log \frac{P(w_k \mid c_i)}{P(w_k)} + P(\overline{w}_k \mid c_i) \cdot P(c_i) \log \frac{P(\overline{w}_k \mid c_i)}{P(\overline{w}_k)} \\ &\quad + P(w_k \mid \overline{c}_i) \cdot P(\overline{c}_i) \log \frac{P(w_k \mid \overline{c}_i)}{P(w_k)} + P(\overline{w}_k \mid \overline{c}_i) \cdot P(\overline{c}_i) \log \frac{P(\overline{w}_k \mid \overline{c}_i)}{P(\overline{w}_k)} \end{aligned} \quad (10)$$

위와 같은 방법으로 임의의 단어 w_k가 클래스c_i에 어느 정도 기여하는가를 측정한다. 이 논문에서는 상호 정보를 이용하여 특징을 선택하도록 하였다.

4. 시스템 구현

본 시스템은 웹을 기반으로 구현되었으며, 웹 브라우저를 통해서 작동한다.



<그림 1 시스템 실행 화면>

4.1 색인 구조

클래스와 클래스가 포함하는 단어, 그리고 관련된 값을 찾기 위한 색인 구조는 배열을 이용하여 구성할 수 있다. 그러나, 이 경우, 단어들을 찾는 과정에서 생기는 시간적 낭비가 크고, 데이터베이스를 이용하는 경우에도 질의문을 처리하기 위해서 많은 시간이 필요하다. 그래서 이 시스템에서는 트라이(Trie)를 기반으로 하는 색인 구조를 사용하였다.

트라이의 키워드로서 단어 이름을 사용하였고, 연결된 데이터 노드는 다음과 같이 구현하였다.

```
typedef struct trienode *trie_ptr;
typedef struct trienode {
    char      key;
    data_ptr data;
    trie_ptr child, next;
};

typedef struct datanode *data_ptr;
typedef struct datanode {
    char    *classname, *word;
    int     tf;
    double mi;
    data_ptr next, c_next;
};
```

<표 1 트라이 노드의 구현>

이 경우 “한 단어가 어떤 클래스들에 속해 있는가?”라는 정보는 매우 쉽게 얻을 수 있지만, “한 클래스에 어떤 단어가 속해 있는가?” 라는 정보는 얻기가 힘들다. 그래서 이 시스템에서는 클래스 이름을 키워드로 하는 트라이 인덱스를 추가하고 단어 이름을 키워드로 갖는 데이터 노드를 공유함으로써 위와 같은 문제를 해결하였다.

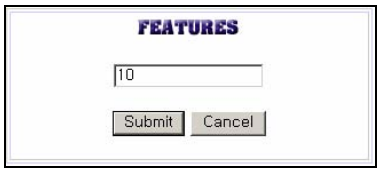
4.2 사용자 개입

베이지안 모형에 여러 독립 가정들을 부여하게 될 경우, TF-IDF에 기반한 모형과 유사한 단점을 가지게 된다. 예를 들면 ‘스포츠’라는 분류 구조에서 ‘선수’라는 단어는 어떤 스포츠 종목에서나 쉽게 등장할 수 있는 단어이다. TF-IDF 모형에서는 이런 원인이 정확도를 떨어뜨리는 주요한 원인이 되었다. 마찬가지로 베이지안 Naïve 모형에서도, 이런 단어들이 가중치를 잃어버리도록 만드는 확률적 인과관계가 무시되었기 때문에 비슷한 결과를 가져다주게 된다.

이 시스템에서는 다음과 같은 인자들에 사용자 개입을 허용함으로써 분류기의 성능을 향상시켰다.

4.2.1 특징 수

특징은 상호 정보값에 의해 정렬된 순서대로 일정한 수만이 고려 대상이 된다. 그 특징 수는 사용자 임의로 결정할 수 있으며, 필요에 따라 증감시킬 수 있다.



<그림 2 특징 수 조정 화면>

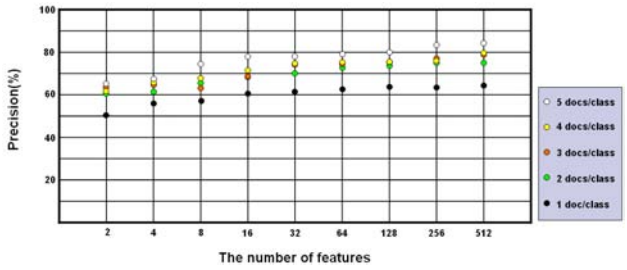
4.1.2 클래스 내 단어 가중치

각 클래스의 단어들은 학습 과정에서 얻는 상호 정보에 의해서 각 클래스에 대한 가중치를 갖게 된다. 그러나, 학습 데이터에 따라서 범용적으로 사용되는 단어가 마치 일부 클래스를 대표하는 것처럼 학습되는 결과가 나타나기도 한다. 이 때 사용자가 적절히 이 단어들의 가중치를 줄여줌으로써 분류 오류를 줄일 수 있다.

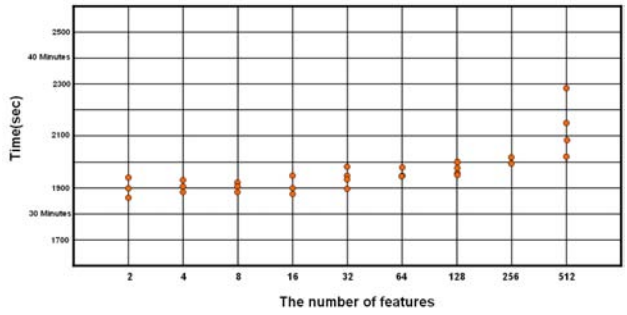
<그림 3 가중치 조정 화면>

5. 실험 결과

시스템의 성능을 측정하기 위해서 98개 클래스를 임의로 설정하고 894개의 문서를 사용하여 실험을 했으며 그 결과는 다음과 같다.



<그림 4 특징 수 대비 정확도>



<그림 5 특징 수 대비 분류 시간>

첫 번째 실험(그림 4)에서 각 클래스마다 사용된 학습 데이터는 100~200개의 단어로 이루어진 문서들을 사용하였다. 클래스별로 비교적 적은 수의 학습 데이터를 사용했음에도 약 80% 정도의 정확도를 보여주었다.

시간에 관련된 실험(그림 5)은 같은 실험을 4회 반복 시행하였다. 그 결과에서 특징 수에 비해 시간 변화가 뚜렷하지 않았던 점은 분류 과정에서 데이터 구조로부터 데이터를 가져오거나, 여러 수치 계산 등 분류 작업 자체에 소요된 시간에 비해서 분류하려는 문서를 디스크와 같은 저장 장치로부터 가져오는데 소요된 시간이 훨씬 크게 작용한다는 것을 반영한다.

6. 결론

본 시스템은 기존의 문서 분류 이론들을 효율적인 자료 구조를 사용하여 구현하였으며, 사용자 개입을 허용할 수 있는 인터페이스를 통해서 학습된 내용을 개선할 수 있다는 장점을 가지고 있다.

적절한 사용자의 개입은 많은 양의 데이터와 시간을 필요로 했던 기존의 시스템에 비해서 시스템 관리자의 노력을 대폭 경감시킬 수 있으며, 이는 문서 분류 시스템을 구현하는데 있어 매우 현실적인 대안이 될 수 있을 것이다.

참고문헌

[1] Fabrizio Sebastiani “Machine Learning in Automated Text Categorization” ACM Computing Surveys Vol.34 No.1 2002

[2] Ricardo Baeza-Yates et al “Modern Information Retrieval” Addison Wesley 1999

[3] Tom Mitchell “Machine Learning” McGraw Hill 1998

[4] Thomas M. Cover et al “Elements of Information Theory” Wiley 1991

[5] Yiming Yang et al “A Comparative Study on Feature Selection in Text Categorization” Proceedings of ICML-97 1997