공학석사학위논문

Hybrid Naive Bayes HMM 기법을 사용한 텍스트로부터의 감정 분류

Emotional Classification from Text Data by Hybrid Naive Bayes HMM

2002년 2월

서울대학교 대학원

컴퓨터공학과

문 현 구

초 록

텍스트 분류의 자동화는 현대와 같이 네트워크 상에서 넘쳐나는 문서들을 빠르고 효율적으로 다룰 수 있도록 해 준다. 그래픽 유저 인터페이스가 발달하면서 채팅이나 메신저 등의 환경에서는 캐릭터를 이용해 사용자에게 보다 재미있고 비주얼한 통신을 제공하려는 노력이 진행되고 있는데, 이러한 노력 가운데 사용자가 입력한 문장에 따라 캐릭터가 그에 상응하는 감정을 표현하게 하기 위한 기술이 연구되고 있다.

본 논문은 텍스트 기반의 통신 환경에서 통계적 학습방법을 사용해 특정 문장에 대한 감정을 분류하는 시스템에 관한 글이다. 채팅이나 메신저 등에서 사용되는 문장의 특징은 사용자들이 맞춤법을 어긋나게 사용하거나 은어나 비어의 사용이 많아 자연언어 분석 기법을 사용하기가 불가능한 경우가 많다는 것이다. 따라서 하나의 어절을 하나의 심벌로 생각하고 통계적으로 구축한 학습 데이터를 사용해 입력 문장을 특정 감정으로 분류하는 기법을 사용한다. Naive Bayes 알고리즘을 사용하는 첫 번째 시스템은 현재 보고 있는 문장의 정보만 가지고 감정을 분류하고 본 연구의 목적인 두 번째 시스템은 HMM에 naive Bayes 알고리즘을 추가한 형태인 hybrid naive Bayes HMM 방법으로서 이전에 본 문장의 정보를 포함해서 입력 문장을 분석한다.

실험을 통해서 hybrid 방법이 순수한 naive Bayes 만을 사용한 시스템에 비해 성능이 높게 나타났음을 알 수 있으며 이모티콘이나 의성어, 물음표 등을 검색하는 전처리기를 추가함으로써 성능이 높아짐을 알 수 있었다. 채팅사이트에서 모은 4만 문장을 학습한 후 1013문장으로 실험한 결과 최종적인 hybrid 시스템에서 93.7%의 정확도가 나왔다.

주요어 : 감정 분류, HMM, naive Bayes

학 번 : 2000-21205

목 차

1 장 서론 1

2 장 관련연구 5

3 장 Naive Bayes 분류자 7

4 장 Hidden Markov Models 10

1 절 Forward algorithm, Backward algorithm
2 절 Viterbi algorithm
3 절 Baum-Welch algorithm

5 장 텍스트로부터의 감정 분류 17

1 절 TexMo1: naive Bayes 분류자를 사용한 감정인식
2 절 TexMo2: Hybrid naive Bayes HMM을 사용한 감정 인식

6 장 실험 및 결과 26

1 절 학습 데이터
2 절 실험 설계 및 결과

7 장 결론 34

1 장

서론

20세기 후반에 들면서 컴퓨터 네트워크는 우리에게 폭발적인 정보와 생활의 편리함을 제공하며 인류에게 없어서는 안 될 도구로 자리 잡았다. 광범위한 정보를 제공하는 것은 물론이고 개인간의 통신수단으로서 컴퓨터 네트워크는 가장 빠르고 효율적 수단이 되었다. 특히 인터넷이 활성화되고 그 사용자의 수가 증가하면서 개인 간 통신 수단으로서의 컴퓨터 네트워크가 기존의 통신망을 대체하는 현상까지 보이고 있다.

근래 들어 네트워크의 데이터 전송 속도가 빨라지면서 음성이나 화상을 통한 통신이 등장하는 추세이나 여전히 가장 자주 사용되는 컴퓨터상에서의 의사 표현 수단은 텍스트이다. 텍스트는 음성이나 영상에 비해 저장 공간을 적게 차지하고 정확도가 높으며 가공이 용이하다는 특성으로 인해 네트워크 환경에서 가장 기본적인 통신수단이 되어 왔다. 이러한 텍스트를 사용하는 대표적인 통신수단으로서 전자메일(e-mail)을 들 수 있으며 그밖에 인스턴스 메신저(instance messenger), 채팅(chatting) 등도 자주 사용되고 있다. 최근의 통신 서비스가 상업화됨에 따라 단지 정보를 전달하는 기능에서 더 나아가 음악을 함께 들려주거나 캐릭터(character)를 이용해 상대방의 관심을 끄는 등 사용자들이 재미있게 즐길 수 있도록 여러 가지 서비스가 제공되는 추세이다.

이렇게 통신 서비스가 다양화됨에 따라 텍스트 기반 통신의 한계 또한 발생하게 된다. 텍스트는 '정보의 전달'이라는 측면에서 볼 때 효과적일 수 있으나 '감정 표현'의 용도로 사용하기에는 이미지에 비해 효율이 떨어진다고 볼 수 있다. 특히 채팅과 같이 주로 개인간의 사적인 대화가 주를 이루는 통신에서는 정보의 정확도보다는 재미를 위한 이미지의 사용이나 사실적인 감정 표현에 중점을 두는 경향이 있는데, 텍스트만으로는 이러한 요구를 충족시키기

힘들다.

텍스트 환경의 이러한 한계를 극복하기 위해 다음과 같이 몇 가지 시도들이 행해지고 있다.

• **이모티콘(Emoticon)의 사용** : 이모티콘(Emoticon)은 채팅 중에 감정을 불어넣기 위해 고안된 감정(Emotion)과 아이콘(Icon)의 합성어로 키보드의 기호, 숫자, 알파벳 등을 적절하게 배합해 주로 얼굴표정을 닮은 기호(Sign)를 만들어 놓은 것이다. 이모티콘의 몇 가지 예가 [표 1]에 나와 있다.

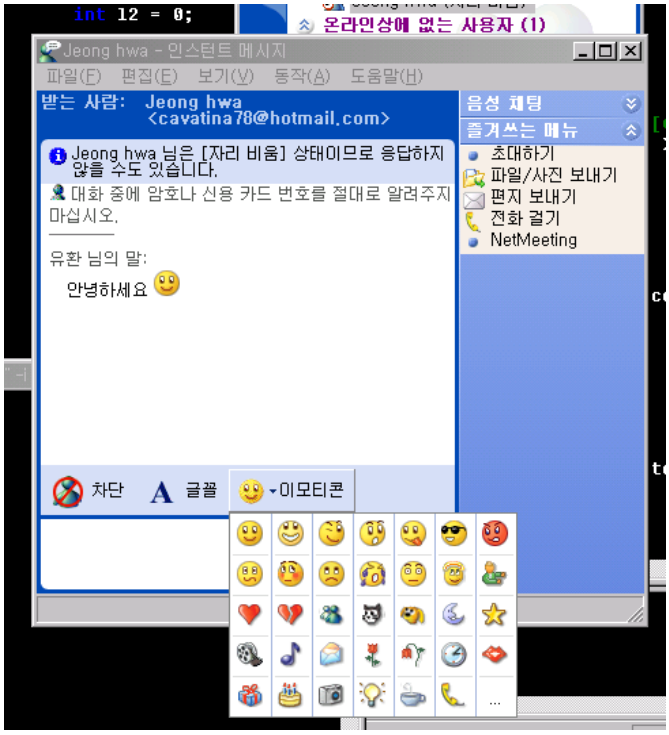
<table>
<tr><td>^^</td><td rowspan="3">가장 기본적인 미소</td><td>: (</td><td>화난 표정</td></tr>
<tr><td>:)</td><td>-.-;</td><td>어이 없음</td></tr>
<tr><td>^_^</td><td>:-P</td><td>메롱</td></tr>
<tr><td>^o^</td><td>박장대소하는 웃음</td><td>-o-</td><td>하품</td></tr>
<tr><td>^^;</td><td rowspan="2">쑥스럽고 멋적은 웃음</td><td>T_T</td><td rowspan="2">울음</td></tr>
<tr><td>^^a</td><td>:'(</td></tr>
</table>

[표 1] 자주 쓰이는 이모티콘의 예

70년대 스마일리(Smiley)라는 이름으로 유닉스(Unix) 전자메일 사용자들을 중심으로 퍼져나간 이모티콘은 개인용 컴퓨터가 보급되면서 전자메일, 인스턴스 메신저, 채팅 등에서 광범위하게 사용되고 있으며 표현할 수 있는 표정의 가지 수 또한 초기에 비해 상당히 증가되었다. 이는 텍스트가 표현할 수 없는 이미지를 일종의 편법을 사용해 해결한 것으로서 컴퓨터 통신 사용자들이 의사 전달 과정에서 이미지를 필요로 한다는 것의 반증이라고 볼 수 있다.

• **화상 통신의 발달** : 컴퓨터 통신망 기술이 발달하고 데이터 전송 속도가 빨라지면서 사용자가 문서를 작성할 필요 없이 스크린을 통해 직접 얼굴을 마주보며 대화할 수 있는 시대가 되었다. 이는 화자의 표정을 직접 전달하므로 텍스트가 가지고 있는 문제를 해결할 수는 있으나 통신선로 및 컴퓨터에 부하를 많이 주고 저장 및 가공의 불편함으로 인해 아직까지는 텍스트에 비해 선호되지 않는 방법이다.

• **감정 표현을 위한 이미지를 선택할 수 있는 인터페이스 제공** : 개인용 컴퓨터가 보급되고 사용자 편의를 위한 그래픽 유저 인터페이스(GUI)가 일반화되면서 채팅이나 전자메일의 작성 방식에도 많은 변화가 있었다. [그림 1]은 한 채팅 사이트에서 제공하는 이모티콘 선택 인터페이스이다. 사용자는 문자 입력창에 원하는 글을 쓴 다음, 특정 이미지의 입력이 필요할 경우 버튼을 눌러 필요한 이미지를 선택한다. 이렇게 선택된 이미지는 자신이 작성한 말과 함께 상대방에게 보여 진다. 이러한 방식은 사용자가 대화 외에도 부수적인 처리를 해 주어야 하는 번거로움이 따른다.



[그림 1] MSN 메신저에서 제공하는 채팅 장면

본 논문은 채팅 등 텍스트 기반의 통신에서 사용자가 입력한 문장을 분석해 해당 감정상태를 자동으로 찾아 주는 방법을 제안한다. 이러한 문제를 해결하기 위한 기본적인 알고리즘은 naive Bayes 분류자와 Hidden Markov

Model(HMM)이다. Naive Bayes 분류자는 미리 학습된 단어의 가중치 정보를 이용해 입력된 문장을 특정 상태로 분류하는데 간단하면서도 비교적 좋은 성능을 낸다[21]. 그러나 이 방식은 현재 입력된 문장만 가지고 판단할 뿐 전후 문맥을 고려하지 않는다는 문제가 있으므로 본 논문에서는 이러한 문제를 해결하기 위해 HMM에 naive Bayes 방법을 혼합시킨 hybrid naive Bayes HMM이라는 방법을 적용하였다.

본 논문의 구성은 다음과 같다. 2장에서 본 연구와 관련된 몇 가지 연구를 소개하고, 3장과 4장에서 감정 분류를 위한 핵심 알고리즘인 naive Bayes와 HMM에 관해 각각 설명할 것이며 5장에서 두 알고리즘이 감정인식을 위해 어떻게 결합되는지에 대한 내용과 실제 시스템을 분석해 볼 것이다. 6장은 실제 채팅 데이터를 대상으로 한 실험에 대한 결과를 제시하고 이를 분석하며 마지막 7장은 결론으로서 연구의 요약과 앞으로의 연구 과제에 대하여 언급할 것이다.

2 장

관련 연구

감정 분류 문제는 텍스트 분류(text categorization) 문제의 특화된 분야이다. 즉 주어진 텍스트를 감정에 따라 분류하는 것이다. 분류(categorization)란 주어진 데이터를 의미 있는 그룹으로 분류하는 방법론으로서, 현대와 같이 인터넷 등의 매체를 통해 텍스트 문서가 넘쳐나는 시대에서 절실한 기술이라 하겠다.

이러한 텍스트 분류 문제를 해결하기 위해 여러 가지 알고리즘이 제안되었다. 회기모델(regression models)이 Yang[5]에 의해 제안되었으며, Cohen[8]의 귀납적 논리 프로그래밍(inductive logic programming), Lewis[10]의 선택 트리(decision tree), Goh[20]의 인공 신경망(neural networks), Joachims[12]의 서포트 벡터 머신 (support vector machine), 그리고 통계적 분류자[9, 16, 21] 등에 대한 연구가 주목할 만 하다.

본 논문에서는 이러한 분류 문제를 해결하기 위해 Hidden Markov Models(HMM)이라고 하는 확률 모델(probabilistic models)을 사용한다.

HMM은 음성인식 분야에서 가장 성공적으로 적용되고 있는데 여기서 HMM의 파라메터는 시간에 따른 음소 단위의 변화를 모델링 하는 천이확률(transition probability)과 각 상태에서 특정한 소리의 스펙트럼(spectrum)이 나타날 확률을 모델링 하는 출력 확률로 구성된다. Rabiner[1]는 파라메터의 학습을 위해 Baum-Welch 알고리즘을 사용하고, 은닉 상태집합의 경로를 찾기 위하여 Viterbi 알고리즘을 사용해 성공적인 음성인식 시스템을 구현하였다.

품사 태깅(Part-of-Speech Tagging) 분야는 HMM의 사용에 대한 아이디어가 본 논문과 비슷한 경우이다. 자연언어 처리 분야에서 문장의 품사를 인식하기 위한 연구는 여러 가지 방법론을 통해 진행되고 있는데, 이 중 HMM을 사용한 품사 태깅은 문장의 품사 연결을 모델의 상태 전이로 보고, 각 품사에서 나올 수 있는 단어가 관찰 데이터라고 생각하며 모델을 학습한다. 이때 HMM의 학습에 대한 접근 방법은 교사학습(supervised learning)과 비교사학습(unsupervised learning)의 두 종류가 모두 가능하다. 교사학습은 미리 사람에 의해 각 단어가 품사 태깅이 되어 통계정보가 구축 된 후에 Viterbi 알고리즘을 사용해서 품사를 분류하는 반면, 비교사 학습은 Baum-Welch 알고리즘을 사용해 사람에 의해 미리 품사가 태깅 되는 비용을 줄일 수 있다. HMM을 사용한 품사 태깅에 관한 연구는 Kupiec[17]등이 있다.

본 연구와 가장 관련 있는 HMM 사용 분야는 Frasconi[4]의 연구이다. 이 연구에서 Frasconi는 HMM의 관찰확률(observation probability)을 구하기 위해 naive Bayes 분류자를 사용하는 hybrid 시스템을 채택했다. 이 연구에서는 OCR 텍스트로 이루어지고 페이지가 나누어져 있는 전자 도서관의 문서를 입력으로 받아 각 페이지를 분류하는데 관심을 두고 있다. 이때 분류 기준은 표지 제목, 목차, 본문, 인덱스 등 책의 구조이다. HMM의 각 상태는 하나의 페이지가 어떤 구조에 속해 있는지와 연관되어 있고, 관찰 심볼은 각 페이지에 포함된 단어나 메타 데이터의 분포를 naive Bayes 분류자를 사용해 모델링한다. 여기서 HMM은 부분적으로 사람에 의해 상태 번호가 매겨진 labeled data에 의해 학습되며 unlabeled data에 대한 학습을 위하여 [3]과 [11]에서 제시한 EM알고리즘을 사용했다. 이 연구에서도 역시 최적 경로를 찾기 위해 Viterbi 알고리즘을 사용했다.

이 문제는 [6, 7]에서 제시한 HMM의 정보 추출(information extraction)문제와 관련이 있는데, [6, 7]에서는 각 단어에 대해 상태를 할당한 반면 이 연구에서는 하나의 페이지를 하나의 상태에 할당했다는 점이 다르다.

3 장

Naive Bayes 분류자

Naive Bayes 분류자는 잘 알려진 전통적인 분류 방법으로서 간단한 구조에 비해 비교적 좋은 성능을 나타내는 분류방법으로서 텍스트 문서 분류에 성공적으로 사용되어 왔다[21].

한 문장이 특정 상태 집합 S가 될 확률 값을 가지는 단어들의 연결로 되어 있는 경우 naive Bayes 분류기는 해당 문장의 상태를 결정할 수 있는데, 이러한 각 단어가 가지는 확률 값은 미리 태깅(tagging)된 학습데이터에 의하여 구성된다. 여기서 상태라 함은 주어진 문장이 분류될 범주를 뜻하고 태깅은 사람이 직접 단어나 문장을 특정 상태로 분류하여 번호를 매기는 일(labeling)을 말한다.

주어진 문장이 어떤 상태에 속하는지 알기 위해 Bayesian식 접근방법을 사용하려면, 해당 문장이 단어들의 연결 $\langle a_1, a_2 \cdots a_n \rangle$으로 이루어져 있다고 가정하고, 다음 식에서 가장 높은 목표함수 값 q_{MAP}을 계산해야 한다.

$$q_{MAP} = \underset{q_j \in S}{\operatorname{argmax}} P(q_j | a_1, a_2 \cdots a_n)$$

위의 식은 Bayes 정리에 의하여 다음과 같이 쓰여진다.

$$\begin{aligned} q_{MAP} &= \underset{q_j \in S}{\operatorname{argmax}} \frac{P(a_1, a_2 \cdots a_n | q_j) P(q_j)}{P(a_1, a_2 \cdots a_n)} \\ &= \underset{q_j \in S}{\operatorname{argmax}} P(a_1, a_2 \cdots a_n | q_j) P(q_j) \end{aligned} \tag{2.1}$$

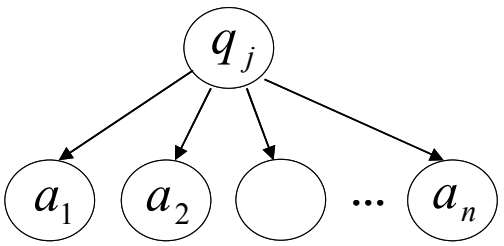
위의 식 (2.1)은 사람이 직접 태깅 한 학습 데이터로부터 구해야 한다.

$P(q_j)$는 단순히 해당 상태 q_j에 속하는 문장이 학습 데이터에서 몇 번 나왔는가를 셈으로써 구할 수 있지만, 특정 문장이 어떠한 상태에서 나올 확률인 $P(a_1, a_2 \cdots a_n | q_j)$는 학습 데이터가 엄청나게 방대하지 않는 한 구하는 것이 불가능하다.

Naive Bayes 알고리즘은 한 문장을 구성하는 각 단어들이 서로 독립적(mutually independent)이라는 가정에서부터 출발한다. 또한 단어가 나타날 확률은 문서 내에서 단어의 위치와도 독립으로 가정한다. 다시 말해 n개 단어로 이루어진 문장 $\langle a_1, a_2 \cdots a_n \rangle$이 상태 q_j에서 나올 확률은 다음과 같이 각 단어들이 상태 q_j에서 나올 확률의 곱이라고 가정하는 것이다.

$$P(a_1, a_2 \cdots a_n | q_j) = \prod_i P(a_i | q_j) \qquad (2.2)$$

이를 Bayesian 네트워크로 나타내면 [그림 2]와 같다.



[그림 2] Bayesian network로 나타낸 naive Bayes 분류자

이러한 가정은 실제 문서의 특성과는 많이 어긋나지만 이러한 단점에도 불구하고 naive Bayes 알고리즘은 텍스트 문서 분류에서 비교적 좋은 성능을 보여주고 있다.

위의 식 (2.2)를 식 (2.1)에 대입함으로써 우리는 다음과 같이 naive Bayes 분류자를 얻게 된다.

Naive Bayes classifier:

$$q_{NB} = \underset{q_j \in S}{\operatorname{argmax}} P(q_j) \prod_i P(a_i|q_j) \qquad (2.3)$$

여기서 q_{NB}는 naive Bayes 분류자의 목표 값을 말한다. 이때 $P(a_i|q_j)$를 구하는 것이 $P(a_1, a_2 \cdots a_n|q_j)$를 구하는 것에 비해 훨씬 작은 학습 데이터를 필요로 함을 알 수 있다.

요약하자면, naive Bayes 학습 방법은 태깅 된 학습 데이터로부터 $P(q_j)$와 $P(a_i|q_j)$를 구하는 학습 단계를 포함하고, 새로운 문장이 입력되면 식 (2.3)을 사용해서 가장 높은 확률을 나타내는 상태 q_j를 선택함으로써 입력 문장을 분류하게 된다. 문장을 구성하는 각 단어들이 서로 독립이라는 가정이 만족하는 한 naive Bayes의 q_{NB}와 식 (2.1)의 MAP는 같다.

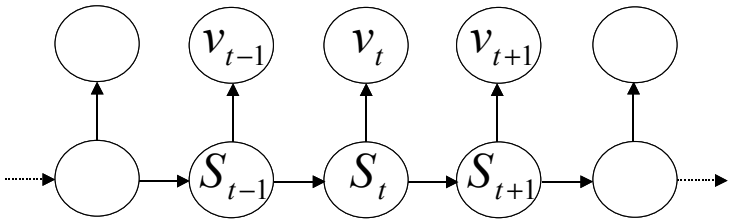
Naive Bayes 학습 방법이 다른 학습 방법에 비해 흥미 있는 점은 그것이 가설 공간을 명시적으로 탐색하지 않는다는 것이다. 그 대신 학습 데이터의 다양한 단어의 빈도수를 셈으로써 가설 공간이 생성된다.

Naive Bayes 분류자는 확률모델이므로 높은 정확도를 위해서 많은 분량의 학습데이터를 필요로 한다. TREC-8에 대해서 실험해 본 결과 충분하고 적합한 문서 데이터를 갖지 않는 경우 좋지 않은 성능을 보였다[24].

4 장

Hidden Markov Models

HMM은 은닉(hidden) 상태를 예측하기 위하여 실제적인 관측을 통해서 변화되는 통계적인 특징들을 확률적으로 모델링 하는 기술이다. 다시 말해 실세계에서 관측되는 어떤 현상들이 특정 상태에서 일정한 확률로 관측되고, 그러한 특정 상태 또한 유한한 집합에서 확률론적 과정(stochastic process)으로 전이될 때 이를 확률적으로 모델링하기 위한 방법론이다. HMM이 가장 성공적으로 적용된 분야는 음성인식(speech recognition) 분야이며 그 밖에도 확률적 품사 태깅(part-of-speech tagging)이나 단백질 분류 문제 등에 적용되고 있다.



[그림 3] Bayesian network로 나타낸 표준 HMM

근래의 HMM을 Bayesian 네트워크의 특별한 종류로 보는 관점[13, 14, 18]은 HMM을 분석하는데 있어 이론적 기반을 제공하였으며 이를 [그림 3]에 나타내었다. 이러한 관점에서 HMM을 특징짓는 구성 요소를 보자면 다음과 같다[1].

• **모델이 가지는 상태(state)의 개수 N** : 이때 상태는 관찰되지 않는 즉 감춰진(hidden)상태이며 모델에 따라 한 상태에서 다른 어떤 상태로의 전이가 가능하거나 또는 어떠한 제약에 의해서 불가능한 전이가 있을 수 있다. 지금부터 상태의 집합을 $S=\{S_1, S_2, \cdots, S_N\}$이라 하고 시간 t에서의 상태를 q_t라 한다.

• **관찰 심볼(observation symbol)의 개수 M** : 각 상태(state)마다 관측되는 심볼(symbol)로서 모델링 된 시스템에서 출력하는 값이 된다. 심볼의 집합은 $V=\{v_1, v_2, \cdots, v_M\}$로 표기한다.

• **상태 전이 확률 (state transition probability)** $A=\{a_{ij}\}$: 상태 i에서 상태 j로 전이할 확률의 행렬을 말하며 다음과 같이 정의된다.

$$a_{ij} = P[q_{t+1} = S_j | q_t = S_i], \qquad 1 \le i, j \le N.$$
$$a_{ij} \ge 0, \qquad \forall i, j$$
$$\sum_{j=1}^{N} a_{ij} = 1, \qquad \forall i$$

만약 상태 i에서 상태 j로의 전이가 일어나지 않을 경우 $a_{ij}=0$이 성립한다. 이를 이용해 우리는 HMM의 topology를 결정할 수 있다. 즉 상태 전이가 일어날 수 있는 경로와 일어나지 않는 경로를 미리 결정해 줄 수 있다.

• **관찰확률(observation symbol probability)** $B=\{b_j(k)\}$: 상태 j에서 심볼 k를 볼 확률의 행렬을 뜻하고 다음과 같이 정의된다.

$$b_j(k) = P[v_k \text{ at } t \mid q_t = S_j], \qquad 1 \le j \le N$$
$$1 \le k \le M.$$

• **초기 상태 확률** (initial state distribution) $\pi=\{\pi_i\}$: 시간 $t=1$에서 각 상태의 확률.

$$\pi_i = P[q_1 = S_i], \qquad 1 \le i \le N.$$

위에서 보듯이 HMM의 완전한 정의는 두개의 모델 파라메터(N, M)와 관찰 심볼의 정의, 그리고 세 개의 확률 측정치 A, B, 그리고 π가 필요하다. 편의상 이러한 파라메터로 구성된 모델을 $\lambda=(A,B,\pi)$로 부를 것이다.

HMM이 실세계에 적용되기 위해서 해결되어야 하는 세 가지 문제가 있는데 이는 다음과 같다[23].

1) Evaluation problem

관찰된 심볼의 시퀀스 $O=O_1O_2\cdots O_T$와 모델 $\lambda=(A,B,\pi)$가 주어졌을 때 그 모델에서 어떻게 관찰된 데이터 O의 확률 $P(O|\lambda)$를 구할 것인가.

2) Decoding problem

관찰된 심볼의 시퀀스 $O=O_1O_2\cdots O_T$와 모델 $\lambda=(A,B,\pi)$가 주어졌을 때 최적의 상태 전이 시퀀스 $Q=q_1q_2\cdots q_T$는 무엇인가.

3) Estimation Problem

가장 큰 $P(O|\lambda)$를 나타내는 모델 파라메터 $\lambda=(A,B,\pi)$를 결정하는 문제.

위의 세가지 문제는 각각 다음에 설명할 forward 알고리즘, Viterbi 알고리즘, Baum-Welch 알고리즘으로 해결이 가능하다[1].

4.1 Forward algorithm, Backward algorithm

Forward 알고리즘은 관찰된 심볼의 시퀀스(sequence) $O=O_1O_2\cdots O_T$와 모델 $\lambda=(A,B,\pi)$가 주어졌을 때 그 모델에서 관찰된 데이터 O의 확률 $P(O|\lambda)$를 구하기 위해 필요하다.

먼저 $\alpha_t(i)$를 다음과 같이 정의한다.

$$\alpha_t(i)=P(O_1O_2\cdots O_t, q_t=S_i\ |\lambda)$$

즉 모델 파라메터 λ가 주어졌을 때 시간 t 에서 시퀀스 $O_1O_2\cdots O_t$를 보고 상태 S_i에 있을 확률이라 한다. 이때 $\alpha_t(i)$는 다음과 같이 귀납적으로 구할 수 있다.

1) Initialization :

$$\alpha_1(i)=\pi_i b_i(O_1),\ 1\le i\le N.$$

2) Induction :

$$\alpha_{t+1}(j)=\left[\sum_{i=1}^{N}\alpha_t(i)a_{ij}\right]b_j(O_{t+1}),\ 1\le t\le T-1,\ 1\le j\le N.$$

3) Termination :

$$P(O\ |\lambda)=\sum_{i=1}^{N}\alpha_T(i).$$

[표 2] Forward 알고리즘의 likelihood 계산 알고리즘

Backward 알고리즘은 Baum-Welch 알고리즘에서 필요한 방법인데

Forward 알고리즘과 비슷한 방식으로 설명되는 개념이므로 여기서 설명한다.

먼저 $\beta_t(i)$를 다음과 같이 정의하면

$$\beta_t(i) = P(O_{t+1} O_{t+2} \cdots O_T | q_t = S_i, \lambda)$$

Backward 알고리즘은 다음과 같이 귀납법을 사용해 설명할 수 있다.

1) Initialization :

$$\beta_T(i) = 1, \quad 1 \le i \le N.$$

2) Induction :

$$\beta_t(j) = \sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j),$$
$$t = T-1, T-2, \cdots, 1, \quad 1 \le i \le N.$$

[표 3] Backward 알고리즘

4.2 Viterbi algorithm

Viterbi 알고리즘은 관찰된 심볼의 시퀀스 $O=O_1O_2\cdots O_T$와 모델 $\lambda=(A,B,\pi)$가 주어졌을 때 최적의 상태 전이 시퀀스 $Q=q_1q_2\cdots q_T$를 찾기 위한 즉, Decoding problem을 해결하기 위한 알고리즘이다.

$\delta_t(i)$를 다음과 같이 정의하고

$$\delta_t(i)= \max_{q_1,q_2,\cdots,q_{t-1}} P[q_1q_2\cdots q_t=i, O_1O_2\cdots O_t \mid \lambda]$$

$\psi_t(j)$를 시간 t 에서 상태 j에 있을 때 $t-1$에 나타날 수 있는 최적 상태를 나타낸다고 할 때 최적 패스는 다음과 같이 구해진다.

1) Initialization :

$$\delta_1(i)=\pi_i b_i(O_1),\ 1\le i\le N$$
$$\psi_1(i)=0.$$

2) Recursion :

$$\delta_t(j)= \max_{1\le i\le N}[\delta_{t-1}(i)a_{ij}]b_j(O_t),\ 2\le t\le T,\ 1\le j\le N$$

$$\psi_t(j)= \arg\max_{1\le i\le N}[\delta_{t-1}(i)a_{ij}],\ 2\le t\le T,\ 1\le j\le N.$$

3) Termination :

$$P^*= \max_{1\le i\le N}[\delta_T(i)]$$

$$q_t^*= \arg\max_{1\le i\le N}[\delta_T(i)].$$

4) 최적 경로 backtracking :

$$q_t^*=\psi_{t+1}(q_t^*), t=T-1,T-2,\cdots,1.$$

[표 4] Viterbi 알고리즘의 최적경로를 찾는 과정

4.3 Baum-Welch algorithm

Baum-Welch 알고리즘은 HMM의 모델 파라메터 $\lambda=(A,B,\pi)$를 학습하기 위하여 사용된다. 이 방법을 기술하기 전에 다음과 같이 $\xi_t(i,j)$와 $\gamma_t(i)$에 대한 정의가 필요하다.

$$\begin{aligned}\xi_t(i,j) &= \frac{\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)}{P(O|\lambda)} \\ &= \frac{\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)}\end{aligned}$$

$$\gamma_t(i) = \sum_{i=1}^{N}\xi_t(i,j).$$

즉 $\xi_t(i,j)$는 시간이 t에서 $t+1$로 흐를 때 상태 전이가 S_i에서 S_j를 나타낼 확률이고 $\gamma_t(i)$는 시간 t에서 상태 S_i를 나타낼 확률을 의미한다.

$P(O^{(k)}|\lambda)$를 간단히 P_k로 표기하고 $O^{(k)}$를 k번째 학습데이터라고 하면 세 개의 파라메터는 다음과 같은 식에 의해 갱신된다.

$$\begin{aligned}\overline{\pi}_i &= \text{expected frequency (number of times) in state } S_i \text{ at time } (t=1) \\ &= \gamma_1(i) \\ \overline{a}_{ij} &= \frac{\text{expected number of transitions from state } S_i \text{ to state } S_j}{\text{expected number of transitions from state } S_i} \\ &= \frac{\sum_{t=1}^{T-1}\xi_t(i,j)}{\sum_{t=1}^{T-1}\gamma_t(i)} \\ \overline{b}_j(k) &= \frac{\text{expected number of times in state } j \text{ and observing symbol } v_k}{\text{expected number of times in state } j} \\ &= \frac{\sum_{t=1}^{T}\gamma_t(j)}{\sum_{t=1}^{T}\gamma_t(j)} \quad s.t.\ O_t=v_t\end{aligned} \tag{4.1}$$

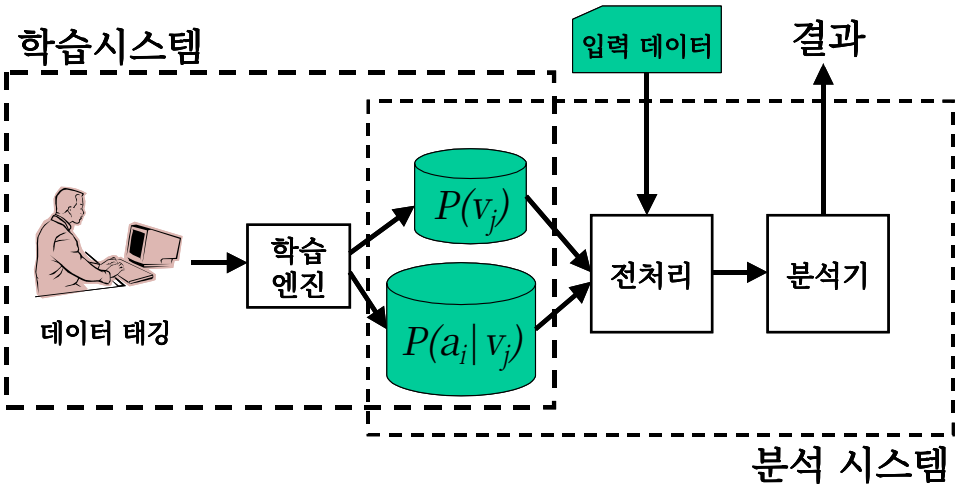
5 장

텍스트로부터의 감정 분류

본 논문에서 제안된 감정인식 알고리즘은 TexMo라는 이름의 시스템으로 구현되었다. TexMo는 사용된 알고리즘에 따라 naive Bayes 분류자가 사용된 TexMo1과 hybrid naive Bayes HMM이 사용된 TexMo2의 두 가지 버전이 존재한다. 지금부터 위 두 시스템과 거기에 사용된 알고리즘에 대하여 설명할 것이다.

5.1 TexMo1 : naive Bayes 분류자를 사용한 감정인식

TexMo1의 시스템 구조도는 [그림 4]와 같다.



[그림 4] 감정 인식 엔진 TexMo의 전체 흐름도

Naive Bayes 분류자의 학습을 위해서는 모아진 데이터를 사람이 직접 각 문장을 특정 상태로 분류해 주는 태깅 작업이 필요하다. 태깅을 위한 상태 결정은 상태간의 모호성이 없고 자주 사용되는 것들이어야 문서의 상태 분류 시 높은 정확도를 기대할 수 있다. TexMo1에서 분류 가능한 감정상태가 [표 5]에 있다.

분류 상태	입 력 예
1. 첫인사	안녕하세요, 하이루, 하2, 안녕
2. 끝인사	빠이, 안녕, 담에봐
3. 미소	^_^ ^^ ^ ^ ^.^ 헤헤, 히
4. 큰웃음	^o^ :-D 하하, 푸하하, 캬캬
5. 킥킥웃음	킥킥, 큭큭, 힉힉힉, 쿄쿄
6. 비웃음	풋, 치~
7. 땀웃음	^^; ^^a 히, 씨익
8. 화남	짜증, 이씨, 제길!
9. 슬픔	T.T ㅠ.ㅠ T_T
10. 울음	흑흑, 엉엉
11. 놀람	:-o 우와, 홧!, 놀라워!
12. 모름	글쎄요, 몰라, 모르겠는데요
13. 지루함	하암~, 졸려, 따분해, 지루하다
14. 사랑해	사랑해, 알라뷰
15. 미안해	미안해, 쏘리, 용서해주세요
16. 축하해	축하합니다, 추카, 생일축하해!
17. 약오르지	:-p 메롱, 부럽지?
18. 질문	뭔데?, 먹었어?, 어디?, 뭐하세여
19. 이해/동의/긍정	글쿤, 아항, 알았어, 그래, 예
20. 거절/부정	시로, 안돼, 싫다, 아니, 아뇨
21. 황당함	황당하군, 허걱
22. 썰렁함	추워, 썰렁하다~
23. 어이없음	--; -.-; -_-; -.-a --a
24. 꾸벅 고개숙임	꾸벅
25. 무동작 상태	아무런 행동을 안하는 상태

[표 5] TexMo1의 분류 가능 범주

[표 5]에서 보듯이 상태3부터 상태7까지 많은 부분이 여러 가지 웃음에 할당되었다. 채팅 데이터를 분석한 결과 가장 사용 빈도가 높은 감정은 상태18 ‘질문’이고 그 다음으로 여러 종류의 웃음의 사용이 많았기 때문이다. 그리고 상태9와 상태23 같은 경우 대화의 분석에서는 잘 나오지 않으나 해당 이모티콘을 사용하는 경우가 많은 이유로 포함되었다.

이렇게 분류 상태가 결정이 되면 여러 채팅사이트에서 모아진 학습데이터가 태깅되어야 한다. [표 6]에 태깅 된 학습 데이터의 일부가 있다.

안녕하세요 @1 안녕하세요 : 1 네, 안냥하세엽. ^^ @1 네 : 19 안냥하세엽 : 1 이싸이트 뭐 하는거에요? @18 이싸이트 : 뭐 : 18 하는거에요? : 18 여기여? @18 여기여? : 18 으엄.. 여기는.. @ 으엄 : 여기는 : ..... 친구를 기다리구 있어요. @ 친구를 : 기다리구 : 있어요 : 아~~ 친구요.. @19 아 : 19 친구요 :	헤헷.. 여자예요.. ^^ @3 헤헷 : 3 여자예요 : 학교 친구. @ 학교 : 친구 : 근데 저별은 뭐에요? @18 근데 : 저별은 : 뭐에요? : 18 얼굴옆에.. @ 얼굴옆에 : 저 별은.. 글쎄요... 방장이라는 표시가 아닐까요? @18 저 : 별은 : 글쎄요 : 12 방장이라는 : 표시가 : 아닐까요? : 18 앙. @19 앙 : 19

[표 6] 채팅 학습데이터의 태깅 예

학습데이터가 태깅 데이터 생성기에 의하여 만들어지면 불필요한 문자들이 제거된 후, 모든 문장 끝에 '@' 표시가 들어가고 각 문장은 어절 단위로 분리되어 ':' 표시가 붙는다. 이러한 학습데이터를 사람이 읽어가면서 '@'가 붙은 문장 끝에 해당 문장이 표현하려고 하는 상태 번호를 붙이고, ':'가 붙은 단어 뒤에 그 단어가 표현하려고 하는 상태 번호를 붙이게 되면 태깅 된 데이터가 완성된다. 아무런 번호를 매기지 않은 단어는 무동작 단어로 간주된다. 문제는 이러한 무동작 단어가 다른 단어에 비해 현저히 많다는 것인데, 이의 해결을 위해서 우선 모든 단어를 다 모은 뒤 특정 횟수 이상 나온 단어만 무동작 단어 사전에 넣어 주었다.

학습엔진은 태깅 된 데이터로부터 식 (2.3)에 있는 naive Bayes 모델에 필요한 파라메터를 측정한다. 즉 '@'로 표시된 문장의 상태 번호를 세서 선험적 확률(prior probability) $P(v_j)$를 구하고 ':'로 표시된 단어들을 태깅 된 상태 번호에 따라 분류해서 $P(a_i|v_j)$를 구한다. 이렇게 모델이 만들어지면 식 (2.3)의 naive Bayes 분류자는 입력되는 문장을 감정상태에 따라 분류할 수 있게 된다. Naive Bayes 분류자를 사용해서 [표 5]의 감정 상태 중 1번부터 24번까지의 값을 계산하고 그중 가장 높은 값을 출력 상태로 결정한다. 이때 선택된 상태의 확률값이 임계값 h를 넘지 않으면 25번 상태인 무동작으로 처리한다.

뒤에 나오는 실험 결과에서 알 수 있듯이 감정인식 시스템의 성능은 전처리의 여부에 따라 크게 달라진다. 채팅데이터에는 불필요한 문자들, 예를 들어 습관적으로 "..."을 붙인다거나 ♡, ♣, ☺ 같은 각종 기호들이 사용되기 때문이 이의 제거가 필요하고, 이모티콘의 사용은 명시적으로 사용자의 감정을 표현하는 것이기 때문에 이를 먼저 검색해서 표현해 주는 등의 처리가 필요하다. 감정인식 시스템에서 감정 분석 알고리즘 외에 사용되는 부수적인 처리는 다음과 같다.

(1) 완성형 코드의 특수문자 제거 : ★②ɑ⅔ 등 감정표현에 사용되지 않는 특수문자 제거.

(2) 이모티콘 검색 : ^^, -.-;, :-) 등의 이모티콘을 검색함. 감정 표현에 있어 이모티콘의 우선순위가 가장 높으므로, 이모티콘이 검색될 경우 감정 분석 알고리즘을 사용하지 않고 바로 감정 코드를 출력한다.

(3) 각종 ASCII 기호 제거 : 이모티콘이 검색된 후 *, #, @, ^ 등 불필요한 ASCII 기호들을 제거.

(4) 웃음 의성어 검색 : 이는 엄밀히 말하면 후처리(postprocessing)로서 감정 분석 알고리즘으로 감정을 찾아 내지 못했을 경우 “하하하”나 “호호호호” 등의 웃음을 나타내는 패턴을 찾아내서 해당 감정을 출력한다. 이는 TexMo1이 띄어쓰기로 이루어진 단어 단위로 통계를 내기 때문이다. 예를 들어 “하하하”와 “하하하하”는 같은 4번 상태 ‘큰웃음’을 나타내는데도 불구하고 다른 단어로 인식되기 때문에 어떤 ‘큰웃음’은 감정분류 알고리즘으로 못 찾는 결과가 발생할 수 있다. 따라서 시스템 뒷단에 웃음 패턴을 인식하는 모듈을 두어야 한다.

(5) 물음표/느낌표 인식기 : 위의 모든 단계를 거치고도 특정 감정을 찾아내지 못했을 경우 마지막으로 물음표가 있으면 상태18 ‘질문’, 느낌표가 있으면 상태11 ‘놀람’으로 분류한다.

지금까지 설명한 처리를 함으로써 naive Bayes 분류자나 다음에 설명한 HMM을 사용한 감정 분류 알고리즘의 성능을 크게 높일 수 있다.

Naive Bayes 분류자는 현재 입력된 문장만으로 감정을 분류하므로 문맥을 고려한 감정 분류를 할 수 없다. 예를 들어 누군가 “그럼”이라는 말을 했을 경우 이 문장은 앞의 문장이 “오늘 5시에 만나는 거 알고 있지?”일 경우 상태19 ‘긍정’이 되고 “그건 내가 한 짓이 아니야!”일 경우 상태18 ‘질문’이 될 수 있다. 이러한 모호성은 앞의 문장에서 정보를 얻지 않을 경우 해결이 불가능한데, naive Bayes 분류자가 가지고 있는 단점 중 하나이다. 이 문제의 해결을 위해 앞뒤 문맥 정보를 사용해 문장을 분석하는 HMM을 도입했다.

5.2 TexMo2 : Hybrid naive Bayes HMM을 사용한 감정 인식

HMM은 관찰 불가능한 상태 전이를 관찰 가능한 심볼을 발생시키는 확률론적 과정을 통하여 모델링(modeling)한다. 따라서 HMM 파라메터를 사용하여 입력된 텍스트를 분석해 해당 텍스트에서 어떻게 상태 전이가 일어나는지를 추측할 수 있다. HMM의 이러한 특징은 naive Bayes 분류자에서 문맥 정보를 얻지 못하는 단점을 해결할 수 있게 해준다. 지금부터는 HMM에 naive Bayes 분류자를 혼합해서 사용한 감정인식 시스템인 TexMo2에 대하여 설명할 것이다.

5.2.1 모델의 구현

HMM 모델에서 상태의 전이는 곧 감정 상태의 전이를 나타낸다. TexMo2에서 분류할 수 있는 감정 상태가 [표 7]에 나와 있다.

분류 상태	용 도
0. 무동작	
1. 밝음	인사, 축하, 미소, 사랑해, 기쁨
2. 어두움	화남, 슬픔, 진지함
3. 의문	질문, 모름
4. 긍정	이해, 승낙, 동의
5. 부정	거절, 반대
6. 흥분	놀람, 강조

[표 7] TexMo2의 분류 가능 범주

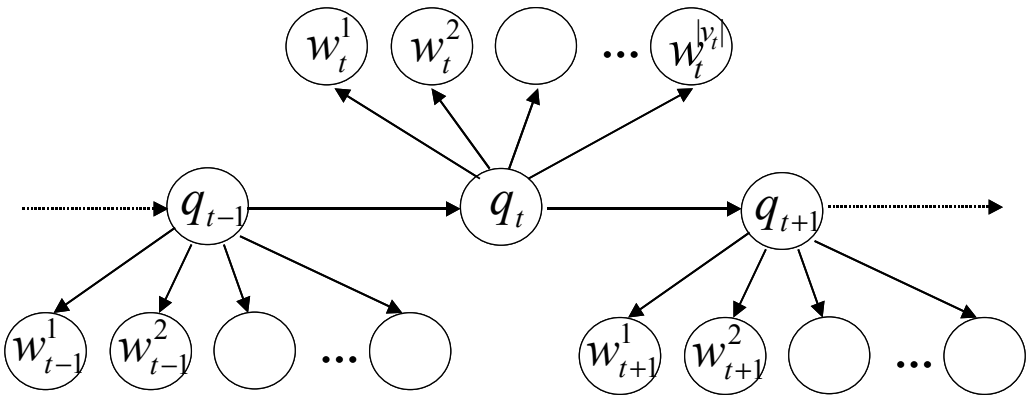
위와 같은 상태는 많은 채팅 데이터를 분석한 결과 뚜렷하게 상태 전이를 나타내고 이미지로 명확히 처리할 수 있는 감정들을 고른 것이다. 위의 감정 상태는 [표 5]의 TexMo1에서 사용하는 감정 상태 집합을 비슷한 유형의 것들끼리 묶어 줬다고 볼 수 있으므로, TexMo1에서 사용한 학습 데이터를 그대로 사용할 수 있었다.

HMM의 관찰 심볼은 한 문장, 즉 일반적으로 한 줄 정도 길이의, 사람이 한번에 입력하는 단어의 시퀀스가 된다. 관찰 심볼이 하나의 심볼, 즉 단어가 아니라 단어가 모인 문장이기 때문에, 본 논문에서는 HMM에 약간의 변형을 가한 방법을 사용한다. 식 (4.1)에 나온 j번째 상태에서 문장 t가 나올 확률인 관찰확률(observation symbol probability) $b_j(t)$를 다음 식과 같이 전체 문장에 대한 naive Bayes 분류자의 각 단어가 나올 조건부 확률로 대체한 것이다.

$$b_j(t) = P(v_t|q_j) = \prod_{i=1}^{|v_t|} P(w_t^i|q_j)$$

여기서 v_t는 t번째 입력한 문장이고 w_t^i는 문장 v_t를 이루는 단어들이다. 이때 $P(w_t^i|q_j)$는 TexMo1에서 사용되었던 것과 같은 태깅 데이터를 naive Bayes 분류자의 학습을 통해 나온 값으로 초기화한다.

위와 같은 모델을 Bayesian 네트워크를 사용하여 표현하면 [그림 5]와 같다.



[그림 5] Hybrid naive Bayes HMM에 대한 Bayesian network

태깅된 학습데이터를 사용해서 갱신되는 HMM 파라메터는 상태 전이확률(state transition probability) a_{ij}이다. 관찰 확률(observation symbol probability)을 저장하는 행렬 $b_j(l)$는 고정시켰다. [4]에서는 EM 알고리즘을 사용하여 $b_j(l)$을 고정시키지 않고 태깅되지 않은 데이터를 사용하여 학습하는 방법을 제안하고 있으나, TexMo2에서는 TexMo1에서 이미 충분한 양의 태깅된 학습 데이터가 구축되었으므로 그러한 처리를 하지 않았다.

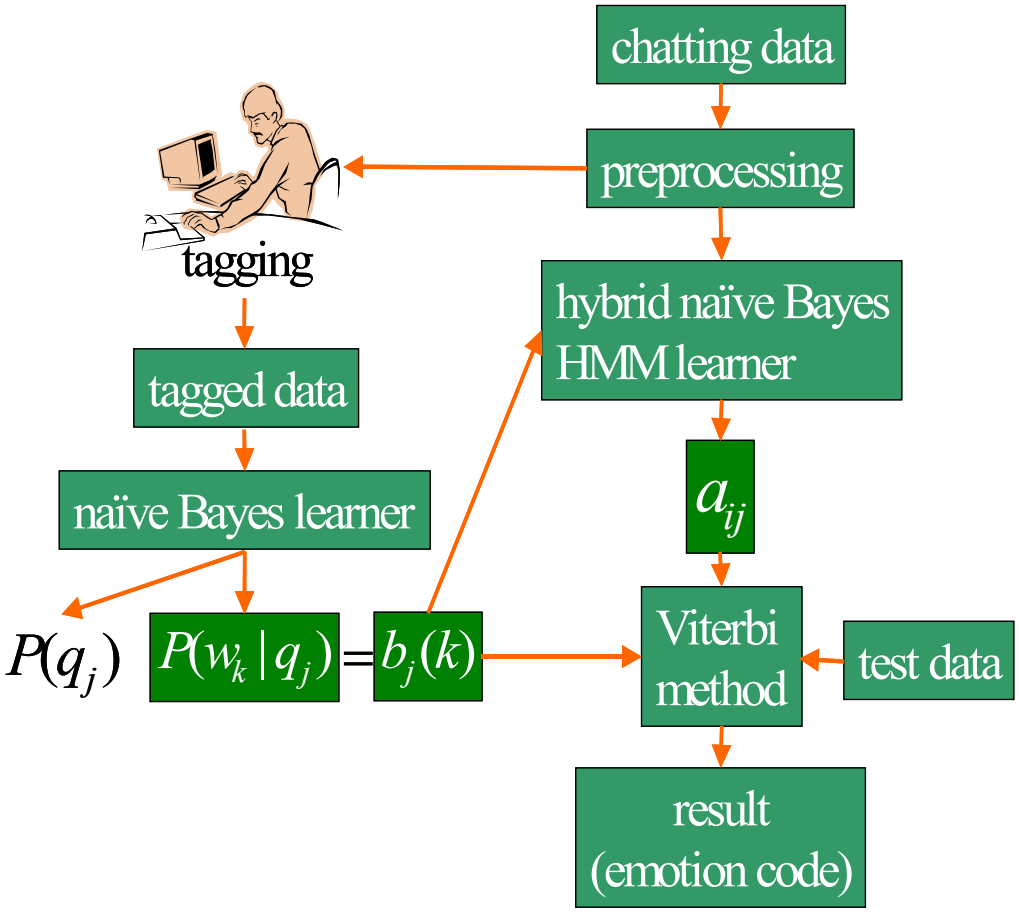
5.2.2. 시스템 구조

[그림 6]은 지금까지 설명한 hybrid naive Bayes HMM을 사용하는 감정 분류 시스템인 TexMo2의 동작을 요약한 것이다. 학습을 위해 모아진 채팅 데이터는 사람이 직접 각 문장에 감정상태 태그를 붙여 주고, naive Bayes 학습기는 이렇게 태깅된 데이터에서 관찰 확률(observation symbol probability) $b_j(k)$를 생성한다. 이 관찰확률은 hybrid naive Bayes HMM 알고리즘을 사용하여 학습 시 사용된다. 이때 관찰확률 $b_j(k)$는 고정되고 HMM의 파라메터 $\lambda=(A,B,\pi)$ 중 나머지 파라메터 A와 π가 학습되어 다음 단계로 보내진다.

학습 데이터의 전체 채팅 데이터를 하나의 시퀀스로 만들어 입력된다. 즉 다른 주제에 대한 대화 내용들이 이어져서 하나의 시퀀스로 입력되는 단일 관찰 (single observation)에 의한 HMM 학습이 사용된다. 시스템을 엄밀히 구현하려면 다중 관찰 (multiple observation)에 의한 HMM을 구현해야 하나 채팅의 특성상 어떤 주제의 대화가 진행되더라도 중간에 다른 사람이 끼어들어 갑자기 새로운 주제로 변환되는 경우가 많으므로, 이러한 노이즈를 감수할 수 밖에 없었다.

테스트를 위해서 채팅에서의 실제 데이터가 입력되면 Viterbi 알고리즘을 사용하여 입력 문장의 감정 상태를 출력한다. 이때 실시간으로 입력되는 데이터의 처리를 위해 이미 나온 N개의 문장을 사용하여 최적 패스를 구한다. 가장 적합한 N은 실험을 통해 찾을 수 있다. 초기에 입력된 문장이 N보다 작을

경우 naive Bayes 방법을 사용해서 출력을 구한다. 그리고 N개 이상의 문장이 들어오기 시작하면 한 문장이 입력될 때마다 이전 N개까지의 문장을 가지고 상태 전이를 예상해서 현재 입력 문장의 감정 상태를 결정한다. 따라서 이 방법을 사용할 경우 현재 입력에 대하여 결정한 상태가 이후의 입력에 따라 변경될 수 있으나 상태 결정에 반영되진 않는다.



[그림 6] hybrid naive Bayes HMM을 사용한 TexMo2의 시스템 흐름도

6 장

실험 및 결과

6.1. 학습 데이터

감정인식 시스템의 학습에 사용되는 데이터는 채팅 문장이다. 4곳의 채팅 사이트에서 다양한 연령대의 대화 내용을 모았으며 전체 문장은 대략 4만 문장 정도이다. 이 문장들은 모두 전처리 후 사람에 의해 정의된 감정상태 번호로 태깅 되었다. [표 8]은 각 상태에 따른 사전의 크기를 나타낸다. TexMo1의 데이터는 상태 분류가 안 되는 경우 무동작 상태로 처리했기 때문에 무동작에 대하여 별도로 사전을 구축하지 않았으나 TexMo2의 경우 무동작 상태도 상태 전이에 포함되어 확률을 계산해야 하기 때문에 사전이 필요하다. [표 8]의 TexMo1 데이터에서 볼 때 18번 질문상태의 단어가 유난히 많은 것을 볼 수 있다. 이는 형태소 분석을 하지 않고 어절 단위로 데이터를 모았기 때문인데, 한국어 특성상 '질문'은 어미에 의해서 표현되는 경우가 많고 그 앞에 붙는 어간은 무한히 변형될 수 있기 때문이다. 예를 들어 '모름'이라는 상태를 나타내는 단어는 "몰라", "모르겠는데", "모르겠다" 등 '모'나 '몰'로 시작되고 어미는 한정되어 있는 반면 '질문'을 나타내는 단어는 규칙을 찾기 힘들다. 그러나 사전을 구축하고 난 결과 '질문'에 해당하는 단어들의 80% 이상의 물음표를 가지고 있는 것을 고려해서 TexMo2의 학습데이터 구축 시 상태3 '의문'에서 물음표를 가지고 있는 단어는 모두 제거했다. 왜냐면 감정 분석 알고리즘이 감정 상태를 못 찾아낼 경우 별도로 물음표를 인식하는 후처리 모듈을 두었기 때문이다.

TexMo1 (naive Bayes)		TexMo2 (hybrid naive Bayes HMM)	
상태	포함 단어 개수	상태	포함 단어 개수
1. 첫인사	310	0. 무동작	3072
2. 끝인사	234	1. 밝음	1162
3. 미소	110	2. 어두움	513
4. 큰웃음	236	3. 의문	969
5. 킥킥웃음	191	4. 긍정	453
6. 비웃음	21	5. 부정	288
7. 땀웃음	5	6. 흥분	134
8. 화남	253		
9. 슬픔	64		
10. 울음	35		
11. 놀람	89		
12. 모름	129		
13. 지루함	72		
14. 사랑해	28		
15. 미안해	80		
16. 축하해	31		
17. 약오르지	33		
18. 질문	5164		
19. 이해/동의/긍정	442		
20. 거절/부정	226		
21. 황당함	33		
22. 썰렁함	29		
23. 어이없음	156		
24. 꾸벅 고개숙임	6		
25. 무동작 상태	0		

[표 8] TexMo1과 TexMo2의 학습데이터에서 각 상태가 가지는 어휘의 개수

채팅 데이터를 학습하는데 있어 가장 특별한 점은 채팅에서 사용되는 문장에 오타나 띄어쓰기가 잘못된 경우가 많다는 것이다. 이는 형태소 분석을 불가능하게 만들 정도로 심각한데 이러한 잘못된 문장을 올바로 고쳐주는 전처리는 하지 않았다. 또한 naive Bayes 방법을 사용할 때는 문제가 되지 않으나 HMM 방식을 사용할 경우 전체 학습 문장에서 대화의 시퀀스 경계 부분, 즉 한곳의 채팅방 대화 내용과 다른 곳의 채팅방 대화 내용이 연결된 부분은 노이즈(noise)로 작용할 수 있다.

6.2 실험 설계 및 결과

실험에 사용된 데이터는 학습데이터를 모았던 채팅사이트에서 비슷한 방법으로 모아진 채팅 문장과 영화 시나리오가 사용되었다. 채팅 데이터와 영화 시나리오의 차이점은 후자가 전자에 비해 맞춤법과 띄어쓰기가 정확하고 표준어에 가깝다는 것이다. 영화 시나리오는 채팅 데이터로 학습된 시스템이 일상대화에 얼마만큼 적용되는지의 비교를 위해 사용되었다. 실험을 위해 각각 채팅 데이터는 1013문장이 사용되었고 시나리오는 420문장이 사용되었다. 이 문장들은 결과 분석을 위해 TexMo1과 TexMo2에서 분류하는 상태별로 각각 미리 태깅 하였다.

[표 9]에 TexMo2의 학습 후 생성되는 hybrid naive Bayes HMM의 상태 전이 확률(state transition probability)을 나타냈다. 쉽게 생각해도 알 수 있듯이 무엇인가를 물어보는 상태인 '의문' 후에 '긍정'이나 '부정'의 상태가 나올 확률이 높다는 것 등이 반영되고 있다.

t / t-1	무동작	밝음	어두움	의문	긍정	부정	흥분
무동작	0.76	0.02	0.0	0.15	0.04	0.01	0.01
밝음	0.47	**0.29**	0.0	0.13	0.07	0.01	0.02
어두움	0.48	0.03	**0.23**	0.17	0.03	0.02	0.03
의문	0.35	0.03	0.0	0.14	**0.25**	**0.22**	0.01
긍정	0.45	**0.25**	0.0	0.15	0.12	0.01	0.01
부정	0.45	0.04	**0.22**	0.17	0.06	0.04	0.02
흥분	0.33	0.15	0.1	0.13	0.03	0.0	0.25

[표 9] Hybrid naive Bayes HMM의 학습 후 시간 t-1에서 t로의 state transition probability의 분포 : 굵은 숫자로 쓰여진 전이 확률을 볼 때 상태 전이가 비교적 예상대로 나오고 있음을 알 수 있다.

[표 10]은 4만 문장의 학습 데이터를 학습한 후 각각의 테스트 데이터에 대한 정확도(accuracy)이다. 정확도는 전처리를 사용하지 않은 순수한 감정분류 알고리즘을 사용했을 때의 결과와 이모티콘, 웃음 의성어, 물음표 등을 찾아내는 전 후처리 모듈을 사용했을 때의 결과로 나누어 나타냈다.

채팅 데이터를 사용한 실험의 결과를 볼 때 hybrid naive Bayes HMM을 사용한 TexMo2에서 전반적으로 성능 향상이 있음을 알 수 있다. 특히 두 가지 경우 모두 전 후처리 모듈을 추가했을 경우 성능향상이 크게 나타났다.

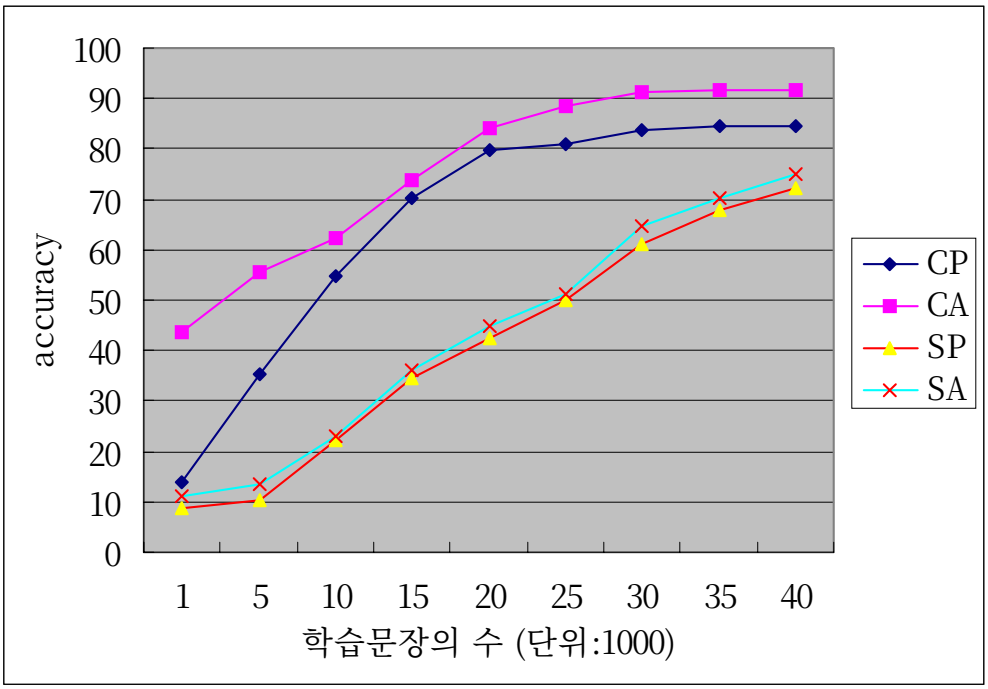
영화 시나리오를 사용한 테스트에서는 특히 TexMo1의 정확도가 전처리를 하지 않을 경우 많이 떨어졌는데, 이는 TexMo1 감정상태가 채팅에서만 주로 사용하는 감정상태로 구성되었기 때문이다. 영화 시나리오의 경우 전 후처리를 추가해도 커다란 성능의 향상은 보이지 않는다. 이는 채팅 데이터에서 성능 향상에 결정적인 요인을 하는 이모티콘의 사용이 영화 시나리오에선 없기 때문으로 분석된다.

		채팅 데이터	영화 시나리오
TexMo1 (naive Bayes)	순수 알고리즘	84.6	72.3
	전,후처리 추가	**91.6**	74.9
TexMo2 (hybrid)	순수 알고리즘	86.9	76.0
	전,후처리 추가	**93.7**	78.1

[표 10] TexMo1과 TexMo2의 정확도

[그림 7]과 [그림 8]은 각각 TexMo1과 TexMo2의 학습 데이터의 양에 따른 정확도의 변화이다. 대략 2만 문장 정도까지 성능이 증가하다가 그 이후부터는 성능 변화가 크지 않는 것으로 보아 채팅에서 사용하는 어휘를 짐작할 수 있다. 특히 이모티콘 인식이 있는 시스템은 사전의 크기가 극히 작더라도 감정 인식률이 높은 편이다. 그만큼 채팅에서 이모티콘의 사용이 잦다는 것을 의미한다. 두 그래프에서 보듯이 다른 알고리즘을 사용해도 결과 그래프는 비

슷한 모양을 하고 있다. TexMo2에서 HMM을 사용하면서 관찰 확률(observation symbol probability)을 위해 naive Bayes를 사용했기 때문에 출력 결과는 비슷한 양상을 나타내는 것으로 보인다.



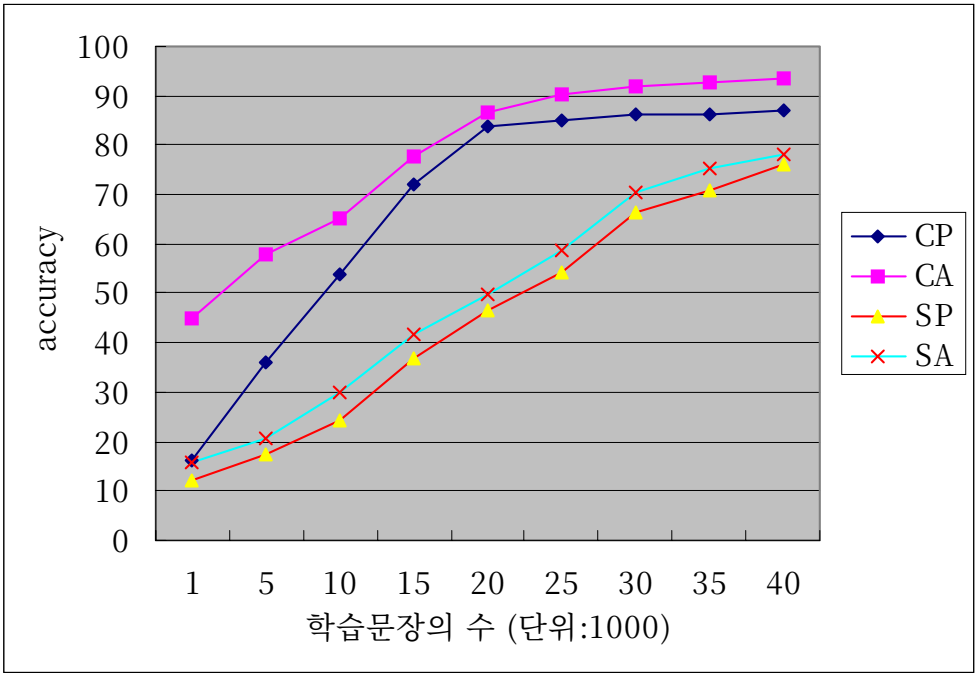
[그림 7] naive Bayes를 사용한 TexMo1의 학습량에 따른 테스트 데이터의 정확도(accuracy) 변화

CP: 채팅데이터를 분류 알고리즘만 사용한 결과

CA: 채팅데이터를 전,후처리를 추가한 시스템으로 분석

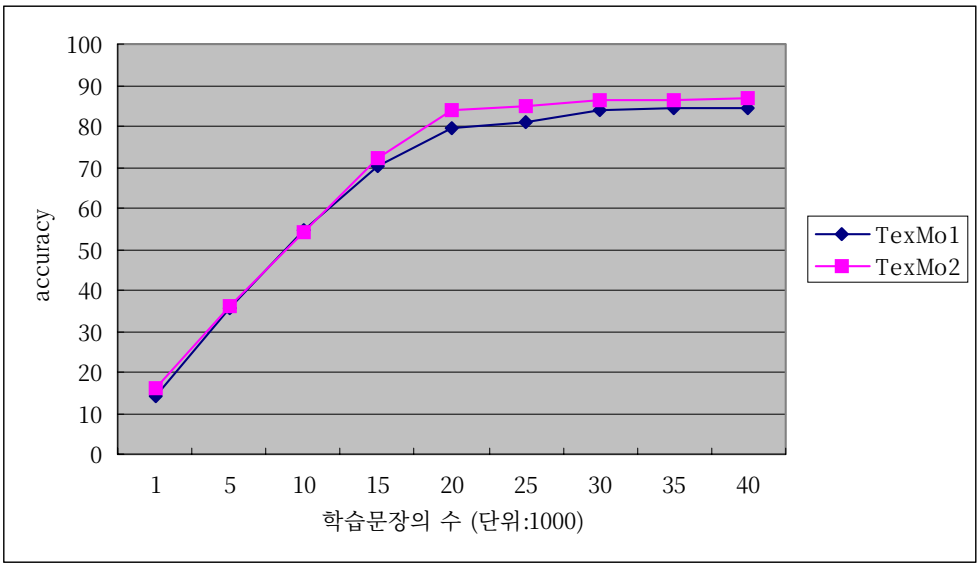
SP: 시나리오를 분류알고리즘만 사용

SA: 시나리오를 전,후처리를 추가한 시스템으로 분석



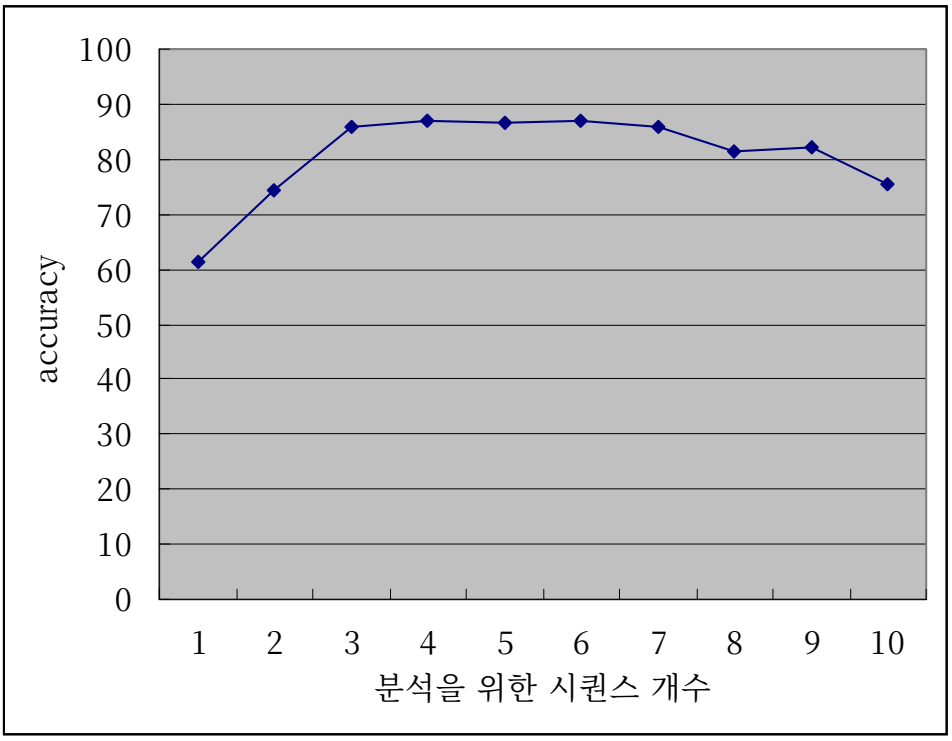
[그림 8] hybrid naive Bayes HMM을 사용한 TexMo2의 학습량에 따른 정확도

[그림 9]는 채팅데이터에 대하여 전 후처리기를 장착하지 않은 두 시스템의 정확도를 비교하기 위한 그래프이다. 전체적으로 TexMo2가 높은 정확도를 보임을 알 수 있다.



[그림 9] TexMo1과 TexMo2의 CP 데이터에 대한 정확도 비교

HMM을 사용해서 실시간으로 채팅 문장을 분석하기 위해 TexMo2는 현재 입력되는 문장 이전의 N개의 문장을 분석한다. [그림 10]은 최적의 N값을 찾기 위해 N을 바꿔가면서 실험한 결과이다. 시퀀스의 길이가 4 이상이 되면 정확도가 점점 감소하는 것을 볼 수 있는데, 이는 채팅의 특성상 대화의 감정 전이가 어느 정도의 길이 이상으로는 일관성 있게 진행되지 않기 때문으로 보인다.



[그림 10] HMM이 분석하는 상태 개수에 따른 정확도 변화

7 장

결론

채팅데이터는 그 특성상 문법적 분석이 불가능하다. 따라서 간단한 naive Bayes 모델을 사용하고 이모티콘이나 의성어 등을 인식함으로써 감정인식에 효과적으로 사용할 수 있다. 실제로 TexMo1으로 실험해 본 결과 좋은 성능을 내는 것을 볼 수 있었다. 그러나 TexMo1의 경우 문맥 정보를 활용하지 못하므로 입력 데이터의 상태 결정이 앞 문장의 상태에 의지해야 하는 경우 분류해 내지 못하는 약점을 가지게 된다. HMM에 약간의 변형을 가한 hybrid naive Bayes HMM을 사용해서 감정인식 문제에 적용해 본 결과 이 알고리즘을 사용한 TexMo2는 TexMo1에서 구분하지 못하던 감정상태를 찾아내어 정확도 향상에 이바지했다.

실험을 통해서 어간과 어미의 결합을 통해 단어가 만들어 지는 한국어의 감정 인식 시스템을 구현하는데 있어 형태소 분석 등의 전처리를 하지 않고도 만족할 만한 정확도를 얻을 수 있음을 알 수 있었다. 이는 특정 감정을 표현하는데 있어 한국어가 특성상 어미변화를 많이 하지 않음을 나타낸다고 볼 수 있다.

띄어쓰기가 어려운 한국어의 특성상 이로 인한 노이즈가 성능저하의 원인 중 하나로 분석된다. 전처리 부분에서 띄어쓰기를 보정해 주거나 자주 발견되는 오타 등을 찾아서 해결해 주는 모듈, 그리고 동의어끼리 모아서 같은 내용으로 인식해 주는 모듈 등이 존재한다면 커다란 성능 향상이 있을 것으로 보이며 이는 앞으로 시스템 업그레이드에 있어서 고려될 사항이다.

참고문헌

[1] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257-286, 1989.

[2] A. Stolcke and S. Omohundro. Hidden Markov Model induction by bayesian model merging. In S. J. Hanson, J. D. Cowan, and C. L. Giles, editors, *Advances in Neural Information Processing Systems*, volume 5, pages 11-18, Morgan Kaufmann, San Mateo, CA, 1993.

[3] K. Nigam, A. McCallum, S. Thrun, and T. Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2/3):103-134, 2000.

[4] P. Frasconi, G. Soda, A. Vullo. Text categorization for multi-page documents: A hybrid naive Bayes HMM approach. *JCDL*, pages 11-20, 2001

[5] Y. Yang and C. Chute. An example-based mapping method for text classification and retrieval. *ACM Transactions on Information Systems*, 12(3):252-277, 1994

[6] D. Freitag and A. McCallum. Information extraction with HMM structures learned by stochastic optimization. In *Proc. 12th AAAI Conference*, Austine, TX, 2000.

[7] A. McCallum, K. Nigam, J. Rennie, and K. Seymore. Automating the construction of internet portals with machine learning. *Information*

Retrieval Journal, 3:127-163, 2000.

[8] W. W. Cohen. Text categorization and relational learning. In *Proceedings of the Twelfth International Conference on Machine Learning,* Lake Tahoe, California, 1995.

[9] D. Lewis and W. Gale. A sequential algorithm for training text classirfiers. In *SIGIR-94*, 1994.

[10] D. Lewis and M. Ringuette. Comparison of two learning algorithms for text categorization. In *Proc. 3rd Annual Symposium on Document Analysis and Information Retrieval,* 1994.

[11] T. Joachims. Transductive inference for text classification using support vector machines. In *Int. conf. on Machine Learning*, 1999

[12] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the European Conference on Machine Learning*. Springer, 1998.

[13] Y. Bengio and P. Frasconi. An input output HMM architecture. In G. Tesauro, D. Touretzky, and T. Leen, editors, *Advances in Neural Information Processing Systems 7*, pages 427-434. MIT Press, 1995.

[14] H. Lucke. Bayesian belief networks as a tool for stochastic parsing. Speech Communication, 16:89-118, 1995.

[15] J. Pearl. *Probabilistic Reasoning in Intellegent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

[16] D. Koller and M. Sahami. Hierarchically classifying documents using very few words. In *Proc. Fourteenth Int. Conf. on Machine Learning*, 1997.

[17] J. Kupiec. Robust Part-of-speech tagging using a hidden Markov

model. In *Computer Speech and Language 6,* 1992.

[18] P. Smyth, D. Heckerman, and M. I. Jordan. Probabilistic independence networks for hidden Markov probability models. *Neural Computation*, 9(2):227-269, 1997.

[19] F. Jensen. *An Introduction to Bayesian Networks*. Springer Verlag, New York, 1996.

[20] H. Ng, W. Goh, and K. Low. Feature selection, perceptron learning, and a usability case study for text categorization. In *Proc. of the 20th Int. ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 67-73, 1997.

[21] T. Mitchell. *Machine Learning*. McGraw-Hill, 1997.

[22] T. Kalt. A new probabilistic model of text classification and retrieval. CIIR TR98-18, University of Massachusetts, 1996.

[23] T. Starner. Visual Recognition of American Sign Language Using Hidden Markov Models. Master's thesis, MIT, Media Laboratory, 1995.

[24] 김유환. Naive Bayes Boosting을 이용한 문서 여과. 석사논문, 서울대학교, 2001.

Abstract

Recently, a huge amount of documents are scattered in the Web. Therefore, automatic classification techniques of text documents help us to organize large 0text documents efficiently. In addition, with a development of graphical user interfaces, there are numerous efforts to satisfy users by injecting those features into chatting or messenger system. Especially, several researchers tried to extract emotional states from text documents.

In this paper, we present an emotional classification system which aims to analyze on-line text chatting document and determine the emotional state of the speaker in each sentence based on statistical learning method. However, even a single sentence in a chatting dialog may contain many misspelled words, jargons, and slang. Therefore, it is extremely hard for us to analyze dialogs through natural language processing techniques. Instead, we try to overcome this problem by transforming each word appeared in the given documents into the corresponding, more abstract, symbol from which statistical data are constructed. With these symbol-based representations, we determine the emotional state of each sentence of documents. Our previous approach uses naive Bayes algorithm which tries to finds emotional states using only the current sentence. The second system, proposed in this paper, is a hybridization of naive Bayes and HMM method which determine emotional states from not only the current sentence but also the immediately preceding ones.

In the experiments, the hybrid method showed a remarkable performance compared with the pure naive Bayes method. In addition, we are able to improve the performance further by attaching a preprocessor which automatically extracts emoticons, onomatopoeic words, question marks, etc which is widely believed to be valuable resources identifying the emotional state of a sentence. Trained with 40,000 sentences, we obtained 93.7% accuracy in the final hybrid system with attached preprocessor for 1013 test sentences.