분할-합병기법을 이용한 HMM 분류기의 적응학습

오수환, 김상운
명지대학교 컴퓨터공학과
e-mail : {ohsh, kimsw}@mju.ac.kr

On Adaptive Learning HMM Classifiers Using Splitting-Merging Techniques

Soo-Hwan Oh, Sang-Woon Kim
Dept. of Computer Science and Engineering
Myongji University

Abstract

In this paper we propose an adaptive learning method for HMM classifiers by using splitting and merging techniques to overcome the problem of the conventional learning, where one HMM classifier per class has been trained, individually. The experimental results demonstrate a possibility that the proposed mechanism could be applied for applications of having multiple clusters in a class.

I. 서론

음성과 같은 시계열패턴에 적용할 수 있는 HMM (Hidden Markov Model) 분류기를 적응적으로 학습하는 방법을 연구하였다. HMM은 통계적 특성과 시간적 구조의 모델링에서 우수성을 가지며 음성과 같은 시계열 패턴을 식별하는데 있어서 가장 성공적인 모델로 널리 쓰이고 있다 [1]. 일반적으로 HMM 분류기에서는 클래스별로 준비된 학습패턴을 이용하여 클래스 수 만큼의 HMM을 학습시키며, 미지패턴이 입력되었을 때 가장 출력확률이 높은 HMM의 클래스로 식별한다. 그러나 음성에서 다른 발음들이 같은 단어로 분류되는 경우처럼 상이한 패턴들이 하나의 클래스에 속하는 경우가 많이 있다. 이처럼 한 클래스에 여러 클러스터들이 비선형적으로 위치할 경우, 클래스간 패턴들이 적절하게 식별되지 않을 수 있다. 본 논문에서는 한 클래스에 하나의 HMM을 설정하는 대신에 학습패턴구조에 따라 적응적으로 학습할 수 있는 새로운 학습법을 제안하였다. 제안법에서는 분할과 합병을 통하여 서로 상이한 패턴은 분리하고, 유사한 패턴끼리는 합병하여 HMM이 패턴구조에 대해 적응적으로 학습될 수 있도록 하였다. 효율적인 분할과 합병을 위해, 각각의 패턴만으로 학습시킨 HMM 출력확률값을 패턴의 고유값으로 정하고, 패턴이 속한 클러스터의 HMM 출력확률값을 그 고유값과 비교한 차이를 분할-합병의 기준으로 삼았다. 기준값에 비해 차이가 큰 패턴은 분할을 통해 새로운 클러스터를 생성하고, 기준값 이하의 차이가 나는 클러스터들은 서로 합병하였다. 실험 데이터로는 ETRI에서 배포한 단어음성 데이터를 이용하였으며, 제안방법의 효용성을 검토한 실험 결과를 보고한다.

이하, II장에서는 HMM 분류기에 대해 설명하고, III장에서는 제안한 HMM 분류기의 적응학습법에 대해 설명한다. IV장에서는 제안한 방법의 학습과정의 실제 음성패턴을 예로 들어, 일반적인 HMM 분류기와 비교하여 인식률과 학습시간의 차이를 고찰한다. 끝으로 V장에서 결론을 맺는다.

II. HMM 분류기

2.1 HMM

HMM (Hidden Marcov Model)은 Markov Process에 기초한 시계열 패턴의 생성과정에 대한 확률적인 모델로서, 여기서 시계열 패턴은 이산적인 매 시간마다 관측되는 기호열 y_1^T이 된다. 따라서 음성과 같은 데이터는 특징을 추출한 후 다시 벡터양자화를 이용하여 일정한 수의 관측기호 중 하나의 값으로 만들어주게 되며 이 과정에서 양자화 오류가 발생하게 되고 이를 줄이기 위해 Continuous-HMM을 사용하게 된다. 이 경우 입력은 각 관측기호에 대한 확률밀도가 된다.

또한 기본적인 HMM분류기의 경우 ML(maximum likelihood)를 사용하기 때문에 변별력이 약한 단점이 있어 ANN과 결합하는 등 여러 대안이 제시되고 있다 [2].

HMM은 유한개의 관측기호와 유한계의 상태(state)로 되어 있으며, 각 상태마다 관측기호를 출력할 확률과 다음 시간에 전이할 상태에 대한 확률, 그리고 초기 상태 확률을 파라메터로 가진다. HMM은 t시간에 상태에서 나온 값 y_t만 알 수 있고 상태 q_t는 알 수 없기 때문에 모든 가능한 상태의 경로에 대해 패턴이 나타날 확률을 구하게 된다.

$$P(y_1^T, q_1^T) = P(q_1)\prod_{t=1}^{T-1} P(q_{t+1}|qt)\prod_{t=1}^{T} P(y_t|q_t)$$

이렇게 관측패턴이 나타날 확률을 계산하는 방법으로는 전향 및 후향 알고리즘 (forward-backward algorithm)이 있다.

2.2 HMM 분류기의 학습

HMM의 학습에는 분석적으로 풀 수 있는 방법은 없으며 0이 아닌 임의의 확률 파라메터 값을 준 후 반복적인 계산을 통해 파라메터 값을 최적이 되도록 변화시켜간다. 초기 파라메터 값에 따라 성공적인 학습이 보장되지 않으므로 초기값을 달리하여 여러번 학습한 후 가장 나은 것을 취하는 작업이 필요하다.

HMM의 파라메터를 추정하는 방법으로 Baum-Welch 학습 알고리즘과 Viterbi학습 알고리즘이 있으며 Viterbi학습 알고리즘은 Baum-Welch학습 알고리즘에 비해 모든 경우의 수를 고려하여 계산하지 않고 빠르게 계산하기 위해 사용되는 알고리즘이다. 본 논문에서는 Baum-Welch 학습 알고리즘을 사용하여 HMM의 학습에 이용하였다.

HMM 분류기는 일반적으로 ML (Maximum Likelihood) [1],[2]법에 따라 각각의 클래스에 대해 하나씩 HMM을 학습시키며, 그림 1과 같이 미지의 입력패턴이 들어왔을 때, 각 클래스에 해당하는 HMM으로 출력확률을 계산한다. 알고자 하는 것은 $P(w_c|y_1^T)$ 이기 때문에 다음의 정리를 이용하여 $P(w_c)$를 곱한 값으로 클래스를 판별할 수 있다.

$$P(w_c|y_1^T) = \frac{P(y_1^T|w_c)P(w_c)}{P(y_1^T)}$$

그림 1.에서는 $P(w_c)$가 모두 같다는 가정에서 출력확률이 최대인 클래스로 패턴이 식별되는 것을 보여준다.

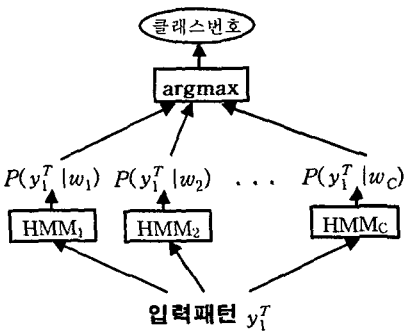

그림 1. HMM 분류기 구조

그런데 실제 응용에서는 하나의 클래스에 여러 상이한 클러스터가 존재할 수 있다. 이런 경우 일반적인 HMM 분류기에서는 적절하게 대응하기가 어렵다.

음성인식의 예를 들면 대표어휘의 맞춤법 표기와는 달리 발음 변이 현상이 자주 발생한다. 이러한 발음 변이된 어휘를 모두 독립된 어휘로 사용하기에는 문제점이 있기 때문에 대표어휘 하나만을 사용하게 되며 모든 발음이 대표어휘에 속하도록 하고 음향모델은 다른 발음에 대해서는 다른 모델을 사용하도록 한다.[4] 그러나 모든 가능한 경우에 대해 예측하여 적절하게 필요한 모델을 만드는 것은 쉬운 일이 아니다.

III. HMM 분류기의 적응학습법

본 논문에서 제안하는 방법은 상이한 패턴이 있는 경우 적절히 클러스터링하여 각기 다른 모델을 가지도록 하는 것이다. 따라서 하나의 클래스에는 여러 개의 클러스터가 생길 수 있으며 각 클러스터마다 HMM을 가진다. 미지의 입력패턴이 들어왔을 때 클래스에서의 출력확률은 클래스의 모든 클러스터에서의 출력확률 중 최대 값이다.

HMM을 적응적으로 학습하기 위하여 다음과 같은 단계를 거친다.

처음에는 일반적인 HMM 분류기와 같은 구조로 시작한다. 그러나 HMM의 학습이 끝나면 가장 상이한 패턴을 하나 찾아 상이한 정도가 미리 정한 임계 값을 벗어나면 새로운 클러스터로 분할이 일어나게 된다. 분할된 클러스터는 새로운 HMM을 가지고 학습된다. 그리고 클래스의 모든 패턴들을 가장 가까운 클러스터로 분배한다. 만약 어떤 두 클러스터에서 패턴들이 상대 클러스터로 들어가더라도 모두 임계 값을 벗어나지 않는 출력확률을 가진다면 두 클러스터는 합병이 일어나게 된다. 클러스터가 변경된 경우 각 HMM의 재학습이 일어난다. 학습이 끝나면 더 이상 클러스터의 변경이 일어나지 않을 때 까지 분할-합병을 반복한다.

HMM의 적응학습 알고리즘은 다음과 같다.

단계 1. 클래스 안의 각 패턴에 대해 1대1로 HMM들을 학습시킨다. 이 때, 각 패턴에 대해 최적으로 학습된 HMM에서의 출력확률을 그 패턴의 고유 값으로 삼는다.

단계 2. 클러스터별로 (초기상태에는 한 클래스가 하나의 클러스터가 됨) 하나의 HMM으로 학습한다. 이 때 고유 값과의 차이가 임계 값 이상으로 낮은 출력확률을 내는 패턴들 중 가장 낮은 출력확률을 내는 패턴을 새로운 클러스터로 "분할 (splitting)"하여 새로운 HMM을 할당한다. 이 때 할당하는 HMM은 단계 1에서 구한 것을 이용하여 학습시간을 절약한다.

단계 3. 각각의 패턴들을 출력확률이 가장 높게 나오는 클러스터로 재배치한 후, 각 클러스터들의 HMM을 새로 학습한다.

단계 4. 단계 3에서 어떤 두 클러스터에서 패턴들이 양쪽 모두에서 임계 값 이내의 출력확률을 내는 경우 두 클러스터를 "합병 (merging)"하고

HMM을 다시 학습한다.

단계 5. 임계 값을 벗어나는 출력확률을 내는 패턴이 없을 때 까지 단계 2부터 단계 4를 반복한다.

클러스터를 분할-합병하는 기준으로 패턴의 고유값을 정하고 그 고유값과의 차이를 삼은 이유는 단순히 출력확률을 기준으로 삼을 경우에는 HMM의 특성상 패턴 자체의 특성에 따라 높고 낮은 출력확률을 내기 때문에 기준으로 삼기 적절치 않다. 예를 들어 단순히 패턴을 짧게 신축하기만 해도 같은 모양의 패턴이 출력확률은 상당히 높아지는 현상이 생기며 패턴 A에 최적으로 학습시킨 HMM에 패턴 B가 더 높은 출력확률을 보이기도 한다. 이런 경우 패턴 B는 패턴 B에 최적으로 학습시킨 HMM에서는 그보다 더 높은 출력확률을 보이므로 식별이 가능하다. 패턴이 클러스터의 중심에 있는 경우 고유값과 같은 출력확률을 낸다고 볼 수 있으므로 고유값과의 차이를 클러스터 중심에서 얼마나 떨어져 있는가로 볼 수 있다.

IV. 실험 및 결과

4.1 실험 방법

실험에 사용한 데이터는 한국전자통신연구소 자동통역 연구실에서 배포한 한국어 음소최적화 단어를 사용하였다. 각 단어는 남자 5명 여자 5명이 각 1번씩 발성하였고 필터링은 LPF 7kHz이고, 샘플링은 16kHz이며, Resolution은 16bit signed인 데이터이다.

이 음성 데이터에서 배포시 표시된 인덱스에 따라 묵음구간을 삭제한 후 Carnegie Mellon 대학에서 배포한 wave2mfcc 프로그램을 사용하여 MFCC (Mel Frequency Cepstral Coefficient)를 추출하였다. MFCC를 추출하는데는 먼저 고주파 성분을 강조해 주는 프리엠퍼시스 (pre-emphasis) 필터를 거친다. 프리엠퍼시스된 신호는 해밍 윈도우를 씌워서 블록 단위의 프레임으로 나누어진다. 이후부터의 처리는 모두 프레임 단위로 이루어진다. 프레임의 크기는 25.6ms를 사용하였고 프레임 이동은 10ms를 사용하였다. 한 프레임의 음성신호는 FFT를 이용하여 주파수 영역으로 변환된다. 주파수 대역을 여러 개의 필터뱅크로 나누고 각 뱅크에서의 에너지를 구한다. 밴드 에너지에 로그를 취한 후 DCT (Discrete Cosine Transform)를 하면 최종적인 MFCC가 얻어진다. MFCC 계수는 12개를 사용하며 이와는 별도로 구한 프레임 로그 에너지가 추가적으로 사용되어 음식인식의 입력으로 사용되는 특징벡터는 13차 벡터가 된다.

다시 추출된 특징벡터들을 가지고 k-means 알고리즘으로 벡터양자화를 수행하였다. k-means 알고리즘은 클러스터의 수를 정한 후, 클러스터의 중심을 적당히 설정하고 거리상 가장 가까운 패턴들을 클러스터에 포함한 후 평균값으로 다시 클러스터 중심을 구하기를 반복한다.

HMM의 입력으로 들어가는 관측 값들에 대한 확률밀도는 입력패턴에서 추출된 MFCC를 k-means 알고리즘으로 구한 양자화 된 값들과 거리의 4제곱에 반비례한 값을 사용하였다.

4.2 실험 및 결과고찰

벡터양자화는 학습에 참여한 패턴들만으로 하였고 HMM의 모든 초기 학습은 3회 반복하여 가장 나은 것을 취했다.

표 1은 두 단어 "가게에"와 "과거에는"을 하나의 클래스로 가정하여 적응학습이 이루어지는 실제 예를 보여준다. "가게에" 단어의 경우 7번째 단어는 "가게에"가 아닌 "과거에" 라는 음성이 잘 못 들어가 있고 1번째 단어와 "과거에는"의 3번째 단어는 묵음구간이 많이 포함되어 있다. 이 실험의 경우 HMM의 state는 12개, 관측기호의 수는 32개를 사용하였다. 분할-합병을 위한 임계 값은 로그를 취한 값으로 30을 사용하였다.

표 1. 적응학습 과정. 여기서 W1은 단어 "가게에"를, W2는 단어 "과거에는"를 나타내고, CNo (1, 2, 3, 등). 클러스터 번호, Diff log(고유값)-log(출력확률)의미한다.

(a)	W1	CNo.	1	1	1	1	1	1	1	1	1	1
		Diff	93.8	63.6	73.1	70.9	38.1	40.1	47.8	41.8	57.9	33.9
	W2	CNo	1	1	1	1	1	1	1	1	1	1
		Diff	52.7	44.8	62.3	51.5	49.8	26.6	66.8	56.9	69.8	39.7
(b)	W1	CNo	2	2	2	2	2	2	1	2	2	2
		Diff	62.7	38.4	43.5	44.7	15.4	28.2	40.6	20.1	41.4	21.6
	W2	CNo	1	1	1	1	1	1	1	1	1	1
		Diff	51.1	36.5	65.8	52.7	39.8	15.4	48.0	37.9	47.1	25.1
(c)	W1	CNo.	2	2	2	2	2	2	1	2	2	2
		Diff	62.7	38.4	43.5	44.7	15.4	28.2	35.8	20.1	41.4	21.6
	W2	CNo	1	1	3	1	1	1	1	1	1	1
		Diff	51.3	35.7	0.0	52.4	38.3	13.4	44.4	33.7	43.4	22.9
(d)	W1	CNo	4	2	2	2	2	2	1	2	2	2
		Diff	0.0	37.6	40.5	44.6	14.5	27.3	48.0	21.4	37.0	21.9
	W2	CNo	1	1	3	1	1	1	1	1	1	1
		Diff	51.3	35.7	0.0	52.4	38.3	13.4	44.4	33.7	43.4	22.9

표 1.(a)에서 1번째 패턴이 고유 값과의 차이가 93.8로 가장 크기 때문에 새로운 클러스터가 할당된다. 표 1.(b)에서는 1번째 패턴에 새로운 클러스터가 할당된 후 1번 클러스터에 속하는 패턴들 중 일부가 2번 클러스터의 HMM에서 더 높은 출력확률을 나타내어 재배치 된 상태를 보여준다. 재배치된 클러스터의 모양을 보면 단어별로 나뉘었으나 "과거에" 와 같이 "과거에는"에 오히려 더 가까운 단어는 "과거에는"으로 클러스터링 되었음을 볼 수 있다.

다시 표 1.(b)를 보면 고유값과의 차이가 가장 큰 것은 W2의 3번째 패턴이 65.8로 가장 크므로 표 1.(c)에서는 분할되어 새 클러스터가 된 것을 볼 수 있다. 3번 클러스터에는 다른 패턴들이 재배치되지 않았다.

표 1.(c)에서는 W1의 1번째 패턴이 62.7로 가장 커서 표 1.(d)에서 4번 클러스터로 새롭게 분할되었고 역시 다른 패턴들이 재배치되지 않았다. 두 패턴은 모두 묵음구간이 많이 남아있어 다른 패턴과는 상이한 패턴이다.

위의 실험 결과는 제안한 학습법이 클러스터링을 잘 하고 있으며 다른 패턴들과 상이한 패턴을 잘 분할시키고 있음을 보여준다. 또한 위의 실험 예에서와 같이 오류가 포함된 데이터에서 오류를 찾아내는 데에도 유용하게 쓰일 수 있는 가능성을 확인하였다.

다음으로, 일반적인 HMM 분류기와 제안한 분할-합병기법을 이용한 적응학습을 하는 HMM 분류기의 인식률과 학습시간을 고찰하기 위해 음성 데이터들 중 인접한 발음의 2음절 단어 114개에 대해 남녀 각 5번씩의 발음을 실험에 사용하였다. state 수는 8개를 사용하였고 관측기호의 수는 32개를 사용하였다. 분할-합병을 위한 임계 값은 로그를 취한 값으로 10을 사용하였다. 실험 결과는 표 2와 같다.

표 2. 일반적인 HMM 분류기와 제안한 방법의 HMM 분류기의 인식률과 학습시간 비교. *여기서 학습시간은 10개 단어를 대상으로 측정한 시간임.

	인식률 (%)	평균 학습시간(sec)*
일반학습	88.97	757
적응학습	92.99	1596

표 2.에서 기존의 학습법과 제안 방법과의 인식률은 큰 차를 보이고 있다. 그러나 이 차이는 학습데이터의 특성에 매우 민감한 값으로, 클래스별로 다수의 클러스터를 갖는 보다 다양한 음성 데이터를 대상으로 실험할 경우 그 차이는 커질 것으로 보여지며, 앞으로의 과제이다.

학습시간의 경우 큰 차이를 보이고 있으며, 이는 제안 알고리즘의 단계 1에서 모든 패턴에 대해 학습을 하기 때문으로 일반적인 방법의 학습시간을 T라고 하면 그 만큼의 시간이 추가로 걸릴 수 있다. 또한 클러스터가 변경될 때 마다 재학습을 하게 되며, 재학습에도 같은 시간이 걸린다고 가정하고 클러스터 변경이 일어나는 회수가 k라고 하면 학습시간은 2T+kT 가 된다. 그러나 학습 데이터가 늘어나는 경우에 k도 늘어나는 것은 아니며 클래스의 성격에 의존하므로 k는 일정하다고 볼 수 있다. 또한 재학습의 경우에는 이전과 동일한 클러스터도 있고 변화된 클러스터도 이미 상당히 학습된 상태에서 학습을 시작하는 것이기 때문에 추가적인 학습시간은 T보다 훨신 적게 걸리게 된다. 따라서 일반적인 HMM 분류기에 비해 일정한 배수 정도의 시간이 걸린다고 볼 수 있다.

실험에서는 컴퓨터에서 실제로 수행되는 시간을 측정하였기 때문에 캐쉬의 성능이나 다른 프로그램의 영향 등에 의해 변화가 생길 수 있으며, 또한 측정 결과 패턴 한 개당 하나의 HMM씩 학습시키는 경우 전체의 패턴을 하나의 HMM에 학습시키는 것에 비해 훨신 적은 학습시간을 보여, 캐쉬의 영향을 상당히 받는 것으로 추정된다. 결과적으로 실제 학습시간은 일반적인 HMM 분류기에 비해 대략 2배 정도의 시간이 걸려 학습시간에 큰 문제점은 없었다.

V. 결론

HMM은 통계적 특성과 시간적 구조의 모델링에서 우수성을 가져 음성인식 등에 널리 쓰인다. 그러나 하나의 클래스로 간주되는 음성데이터에는 여러 형태가 있을 수 있으며, 실험 결과 이런 경우 제안한 방법으로 클러스터링하며 적응적으로 학습하는 것이 효용성이 있음을 알 수 있었다.

본 논문에서는 한 단어에 대해 남녀 5명씩 10개의 발성만을 가지고 실험하였으나, 더 많은 데이터를 가지고 실험한다면 그 결과는 더욱 향상될 것으로 사료된다.

또한 학습시간에서 일반적인 HMM 분류기에 비해 일정한 배수 정도의 시간이 걸리기 때문에 앞으로 극복할 수 있는 문제로, 불필요한 학습 단계를 줄인다면 더욱 개선할 여지가 있는 것으로 보인다.

참고문헌

[1] 김상운, MATLAB으로 배우는 패턴인식 및 학습, 홍릉과학출판사, 2003.

[2] 김상운, 오수환, "HMM-Net 분류기의 학습", 대한전자공학회논문지, 제34권, C편, 제9호, pp. 703-709, 1997.

[3] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition", Proceedings of the IEEE, vol. 77, no. 2, pp. 257 - 286, Feb. 1989.

[4] 박준, "대화체 음성 자동통역 기술", 대한전자공학회지, 제30권, 제7호, pp. 29-38, 2003.

[5] S. Matsuda, M. Nakai, H. Shimodaira and S. Sagayama, "Speech recognition using asynchronous transition HMM", IEICE Transa., vol. E83-D-II, no. 6, pp. 741 - 754, Jun. 2003.