# 대용량 온라인 한자 인식을 위한 클러스터링 거리계산 척도

## (Distance Measures in HMM Clustering for Large-scale On-line Chinese Character Recognition)

김 광 섭 † 하 진 영 ††
(Kwang-Seob Kim) (Jin-Young Ha)

**요 약** 은닉 마코프 모델(Hidden Markov Model: HMM)에 기반을 둔 온라인 한자 인식에서 클래스의 수가 대용량일 경우에는 인식에 걸리는 시간 증가가 좋은 인식 시스템을 구현하는데 있어서의 걸림돌이 된다. 본 논문에서는 이러한 인식 속도 문제를 해결하고자 HMM을 클러스터링하여 인식 속도를 개선하는 방법과 이에 적합한 효율적인 HMM 간의 거리계산법을 제안한다. 유니코드 한·중·일 통합한자로 정의된 총 20,902개의 한자에 대한 온라인 한자 인식 시스템을 구축하는 실험에서 약 2배 정도로 인식속도가 향상됨을 확인할 수 있었고 클러스터링을 하지 않았을 때보다 0.9%의 인식률만 하락한 95.37%의 10순위 인식률을 달성했다.

**키워드** : 클러스터링, HMM 거리계산법, 대용량 클래스

***Abstract*** One of the major problems that prevent us from building a good recognition system for large-scale on-line Chinese character recognition using HMMs is increasing recognition time. In this paper, we propose a clustering method to solve recognition speed problem and an efficient distance measure between HMMs. From the experiments, we got about twice the recognition speed and 95.37% 10-candidate recognition accuracy, which is only 0.9% decrease, for 20,902 Chinese characters defined in Unicode CJK unified ideographs.

**Key words** : Clustering, Distance measure between HMMs, Large classes

## 1. 서 론

은닉 마코프 모델(Hidden Markov Model, 이하 HMM)은 1970년대부터 시작으로 지금까지 문자 인식, 음성 인식, 동작 인식 등 다양한 분야에 적용되어 좋은 결과를 나타내고 있다[1]. 문자 인식은 1980년도 후반부터 HMM을 적용한 연구가 활발히 진행되어 왔고 현재에 이르기까지 좋은 연구 성과가 많이 발표되었다[2,3]. 또한, 연구 목적뿐만 아니라 상당수의 상업용 시스템에서도 사용되어 좋은 결과를 보이고 있다. 이러한 HMM을 적용한 다양한 분야의 좋은 성과는 HMM의 효과적인 모델링 능력과 Baum-Welch 알고리즘[4,5] 같은 강력한 EM(Expectation Maximization) 알고리즘과 인식을 위한 Viterbi 알고리즘[6]이 큰 기여를 했다.

HMM을 적용한 여러 분야 중에 한자를 대상으로 하는 온라인 문자 인식의 경우는 한자의 특성상 획수가 많고 다른 언어에 비해 변형이 심하다. 또한, 클래스의 수가 방대하기 때문에 계산 자원 부족, 인식에 걸리는 시간의 증가 등의 다양한 문제점의 해결을 필요로 한다. 이러한 문제를 극복하기 위해서 대용량 온라인 필기 한자 인식을 위한 HMM 기반의 클러스터링 방법과 이에 필요한 거리계산법을 제안한다.

본 논문에서 제안하는 거리계산법은 미지의 관측 심볼 열에 대해 어떠한 두 HMM의 출력확률이 얼마나 유사할지 비교하는 것이다. 이러한 목적에 부합하

† 학생회원 : 강원대학교 컴퓨터정보통신공학과
mrkwangsub@gmail.com
†† 정 회 원 : 강원대학교 컴퓨터학부 교수
jyha@kangwon.ac.kr
(Corresponding author임)
논문접수 : 2009년 3월 3일
심사완료 : 2009년 7월 6일


정보과학회논문지: 소프트웨어 및 응용 제36권 제9호(2009.9)

기 위해서 HMM 출력확률을 구하는 전향 알고리즘(Forward Algorithm)에 초점을 두어 고안하였고, 제안한 거리계산법을 기존의 다른 연구를 통해 고안된 거리계산법과 비교평가 하였다.

HMM의 클러스터링은 인식대상인 유니코드에 한·중·일 통합한자로 정의[7]된 20,902 개의 한자 클래스에 대해서 각각의 HMM을 생성한 후에 같은 상태 수를 기준으로 구분하여 별개로 클러스터를 구성한다. HMM을 기반으로 클러스터를 구성하기 때문에 클러스터의 센터 값은 클러스터를 대표하는 HMM이 된다. 미지의 입력 데이터를 인식하는 과정에서 모든 HMM에 대하여 출력확률을 구하는 대신 클러스터 센터 값. 즉, 클러스터 대표모델과의 출력확률을 구한다. 이렇게 구해진 출력확률은 정인식에 해당하는 HMM이 포함된 것으로 기대되는 클러스터의 범위를 판단하는 척도로 사용하여 모든 HMM에 대해 출력확률을 구하는 것을 피한다. 이러한 과정을 통해 HMM 출력확률을 구하는 횟수가 클러스터링 비기반의 시스템보다 더 작다. 이는 곧 인식에 걸리는 시간과도 직접적인 연관이 된다.

본 논문의 구성은 다음과 같다. 1장의 서론에 이어 2장에서는 관련연구를 살펴보고 3장에서는 본 논문에서 제안하는 두 HMM 간의 거리계산법에 대하여 설명한다. 이어서 4장에서는 HMM을 클러스터링하여 인식 시스템을 구성하는 방식과 인식과정에 대해서 하고, 5장에서는 본 논문에서 제안한 거리계산법을 이용해 구현한 클러스터링 기반 시스템의 성능 평가 및 분석을 한다. 끝으로 6장에서는 결론을 맺는다.

## 2. 관련 연구

다양한 목적으로 논의된 HMM 간의 거리계산은 주로 KLD(Kullback-Leibler Divergence)[8]방식을 사용하였다. KLD는 두 확률 분포 간의 유사성을 비교하는 방식으로 상대적 엔트로피 측정이라고도 한다. 변수 $x$에 대한 이산적인 확률 분포 $p(x)$, $q(x)$가 있을 때 KLD는 다음과 같이 정의된다.

$$KLD(p(x):q(x)) = \sum_{x\in X}\left(q(x)\log\left(\frac{q(x)}{p(x)}\right)\right) \quad (1)$$

KLD는 대칭적이지 않고 삼각 부등식을 만족하지 않기 때문에 두 확률 분포 사이의 실제 거리는 아니다. 그럼에도 상대적 엔트로피, KLD를 두 확률 분포 사이의 '거리' 개념으로 생각하는 것이 일반적으로 유용하다.

$$KLD(p(x):q(x)) \neq KLD(q(x):x(x)) \quad (2)$$

위와 같이 KLD가 대칭이 아니므로 다음처럼 계산법을 수정하여 사용하기도 한다[9].

$$KLD'(p(x):q(x)) = KLD(p(x):q(x)) + KLD(q(x):p(x)) \quad (3)$$

이때 log 항의 값이 1보다 작아지지 않도록(log 값이 음수로 나타나지 않도록) 식 (1)에 적절한 상수 값 $\alpha$을 더해준다.

$$KLD(p(x):q(x)) = \sum_{x\in X}\left(q(x)\log\left(\frac{q(x)}{p(x)}+\alpha\right)\right) \quad (4)$$

KLD를 사용하여 두 개의 HMM $\lambda$, $\xi$에 대한 거리계산법은 크게 두 가지가 있다. 첫 번째는 MCM(Monte Carlo Method)에 기반을 둬 충분히 임의적인 관측 심볼 열을 생성하여 그에 대한 HMM 출력확률을 비교하는 것이고[10,11], 두 번째는 관측 심볼 열에 의존하지 않고 HMM의 파라미터의 확률 간의 차이를 비교하는 방법이다[12]. 전자에 해당하는 MCM에 따라서 KLD를 적용한 방식은 비교 대상이 되는 HMM의 은닉 상태 수와 관측 가능한 심볼의 수가 많을 때는 현실적으로 적용하기가 어렵다. 예를 들어, 어떠한 HMM의 은닉 상태 수 $S$가 40이면 이 HMM이 표현하는 클래스는 평균적으로 관측되는 심볼 열의 길이가 40이라는 뜻이다. 여기에 관측 가능한 심볼의 종류 $M$이 32이라고 하면 특정한 관측 심볼 열이 아닌 충분히 다양한 관측 심볼 열을 사용하여 정밀한 결과 값을 얻어야 하는데, 이를 위해서는 이상적으로 $32^{40}$개의 관측 심볼 열을 생성하고 이에 대한 출력확률을 비교하는 방식이 된다. 임의로 생성하는 관측 심볼 열의 수를 크게 줄여서 사용할 수도 있지만, 만족스러운 결과 값을 얻으려고 사용되는 양은 여전히 많다.

## 3. HMM 간의 거리계산법

온라인 문자인식에 많이 사용되는 왼쪽-오른쪽(left-to-right) 구조 HMM은 그림 1과 같은 형태를 보인다. $N$는 은닉 상태 수, $a_{ij}$는 은닉 상태 i에서 j로의 전이 확률을 의미한다. 여기서 점선으로 표시된 전이는 관측 심볼 열없이 전이할 수 있는 널 전이를 말한다.

이러한 구조를 갖는 HMM 간의 '거리'는 거리계산법을 적용하려는 분야나 목적에 따라서 그 정의가 달라질 수 있다. 본 논문에서는 사용자로부터 얻어진 관측 심볼 열에 대해서 어떠한 두 HMM이 얼마나 유사

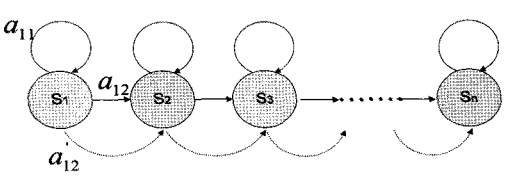


그림 1 왼쪽-오른쪽(Left-to-right) 구조 HMM

한 출력확률을 갖는지를 판단하는 척도로 '거리'를 사용한다. 즉, 두 HMM의 거리 값이 작을수록 임의의 관측 심볼 열에 대해서 기대되는 출력확률이 더 비슷할 것이라고 기대한다.

### 3.1 HMM의 파라미터를 사용한 KLD 거리계산법

HMM의 파라미터를 사용한 거리계산법은 2절에서 알아본 MCM을 기반을 둔 것과 비교해 은닉 상태 수나 관측 가능한 심볼 수에 비례해 계산량이 크게 늘지 않는 것이 장점이다.

본 논문에서 제안하는 HMM 간의 거리계산법은 HMM의 파라미터 값만을 사용하고, 거리계산의 대상이 되는 두 HMM이 서로 상태 수가 같아야 한다는 제약 조건을 갖는다. HMM의 파라미터 값만을 사용할 때 MCM에 기반을 두었을 때보다 계산량이 크게 줄어들고, 서로 다른 상태 수를 갖는 HMM 간의 거리계산을 고려하지 않음으로써 거리계산법이 복잡해지는 문제점을 피할 수 있기 때문이다. 다음은 몇몇 기존 연구에서 두 HMM λ, ξ 간의 거리 값을 KLD 방식으로 HMM 파라미터 확률 값을 비교해 구한 경우다[12,13].

$$D_I(\lambda:\xi)=\sum_i^N\sum_j^N a_{ij}^{\lambda}\log\left(\frac{a_{ij}^{\lambda}}{a_{ij}^{\xi}}\right)+\sum_i^N\sum_v^M b_{iv}^{\xi}\log\left(\frac{b_{iv}^{\xi}}{b_{iv}^{\lambda}}\right) \quad (5)$$

여기서 $A=[a_{ij}]$는 $N*N$의 전이확률 행렬이고 $B=[b_{iv}]$는 $N*M$의 출력확률 행렬이다. 위 방식은 전이확률과 관측 심볼 출력확률을 별개 선상으로 두고 각각 KLD 방식을 통해 계산하므로 부자연스럽다. 왜냐하면, 실제로 HMM 출력확률을 계산할 때 은닉 상태 간의 전이확률과 전이된 상태에서의 관측 심볼 출력확률이 곱으로 표현되기 때문이다. 다시 말해, 전이확률과 관측 심볼 출력확률은 서로 밀접한 관련이 있다. 본 논문에서는 이러한 특성을 반영하기 위해서 KLD를 사용한 식 (5)를 다음과 같이 수정하여 사용하였다.

$$D'_{KLD}(\lambda:\xi)=\sum_s^N\sum_v^M(a_{s-1,s}^{\lambda}\cdot b_{s,v}^{\lambda})\log\left(\frac{a_{s-1,s}^{\lambda}\cdot b_{s,v}^{\lambda}}{a_{s-1,s}^{\xi}\cdot b_{s,v}^{\xi}}+\alpha\right) \quad (6)$$

위 식은 식 (5)에서 전이확률과 출력확률을 따로 계산하는 문제점을 해결하기 위해 두 확률을 곱의 형태로 표현하였다. 이러한 수정은 HMM 출력확률을 계산할 때 가장 기본이 되는 은닉 상태 간의 전이가 관측 심볼 출력 없이 고려할 수 없다는 점을 반영한 것이다. KLD식이 대칭이 아니기 때문에 최종적으로는 다음과 같이 사용한다.

$$D_{KLD}(\lambda:\xi)=D'_{KLD}(\lambda:\xi)+D'_{KLD}(\xi:\lambda) \quad (7)$$

### 3.2 전향 알고리즘에 기반을 둔 거리계산법

HMM 출력확률은 일반적으로 전향 알고리즘을 통해 구해진다. 그렇다면, 두 HMM 간의 출력확률을 비교하는 거리계산법도 전향 알고리즘에 기반을 둔다면 더 효율적인 거리계산이 될 것이라 기대할 수 있다. 여기서 거리계산법을 논하기 전에 전향 알고리즘에 대해서 살펴볼 필요가 있다. HMM $\lambda$가 있을 때 시간 $T$의 흐름에 따라서 관측된 심볼 열 $O=o_1,o_2,\cdots,o_T$가 주어진다면 다음과 같은 전향 알고리즘을 통해 $\lambda$의 출력확률 $P(O|\lambda)$을 구할 수 있다.

*Initialization* (8)

$$a_i(1)=\pi_i b_i(v_i) \qquad 1\le i\le N$$

*Induction*

$$a_j(t)=\sum_{i=1}^N a_i(t-1)a_{ij}b_j(o_t) \quad 1\le j\le N, t=2,3,\ldots,T$$

*Termination*

$$P(V|\lambda)=\sum_{i=1}^N a_i(T)$$

위 식에서 $a_j(t)$는 관측 심볼 열 $o_1,o_2,\cdots,o_t$을 관측하고 시간 $t$에 상태 $j$에 있을 확률이다. 이 확률은 바로 이전 상태인 시간 $t$-1에 존재하는 모든 상태 $i(1\le i\le N)$에서 상태 $j$로의 전이확률에 시간 t에서의 관측 심볼을 출력할 확률 $b_j(o_t)$를 곱하고 다시 이것들을 모두 합한 것이다. 그러므로 시간 t에 상태 j에 있을 확률은 바로 이전 상태까지의 확률과 현 상태로의 전이확률과 현재 상태에서의 특정 관측 심볼의 출력확률만을 고려하면 된다. 이는 1차 마코프 소스(1st order Markov source)의 특성이다. 그러면 두 HMM $\lambda$, $\xi$에 대하여 미지의 관측 심볼 열 $O$에 대한 출력확률을 비교하는 것은 모든 상태에서의 전이확률과 특정 관측 심볼을 출력할 확률을 곱한 값을 비교하는 것으로 고려해 볼 수 있다.

$$D_{FWD}(\lambda:\xi)=\sum_s^N\sum_v^M|a_{s-1,s}^{\lambda}\cdot b_{s,v}^{\lambda}-a_{s-1,s}^{\xi}\cdot b_{s,v}^{\xi}| \quad (9)$$

식 (8)은 은닉 상태 간의 전이 확률 $a$와 관측 심볼의 출력확률 $b$를 곱함으로써 전이확률과 출력확률이 결합하였고, 같은 상태로의 전이 확률에 같은 관측 심볼에 대한 출력 확률을 비교하는 것이다. 이식은 HMM 출력확률을 결정짓는 두 가지 중요한 파라미터 확률을 사용하여 은닉 상태 간의 전이를 자연스럽게 표현하면서 계산식이 복잡하지 않고 $O(N)$의 시간 복잡도를 갖기 때문에 은닉 상태 수나 관측 가능한 심볼 수에 따라서 계산량이 크게 늘지 않는 장점이 있다.

실험 과정을 통해 기존 논문에서 제안했던 거리계산법 $D_I$와 본 논문에서 제안한 $D_{KLD}$, $D_{FWD}$를 사용한 클러스터링의 결과를 비교 평가하겠다.

## 4. 클러스터링과 인식과정

이 절에서는 3절에서 살펴본 거리계산법을 클러스터링에 적용하여 클러스터 모델을 생성하고, 이를 인

식시스템에 적용하고 나서 입력 관측 열에 대해 인식을 하는 과정에 대해서 설명한다.

### 4.1 클러스터링

클러스터링은 비지도학습(Unsupervised learning)으로 샘플집합을 의미 있는 집단(Subgroup)들로 분류하여 데이터 분석, 시각화, 압축 및 전처리와 관련된 많은 분야에서 널리 응용되고 있다.

본 논문에서 샘플이란 인식 대상이 되는 한자 클래스를 개별 단위로 생성한 HMM을 말하고, 의미 있는 집단이라 함은 한 클러스터의 샘플들이 서로 다른 클러스터들의 샘플들에 비해 임의의 관측 심볼 열에 대해서 더 비슷한 HMM 출력확률을 보이는 것이라고 정의한다. 클러스터링 과정에서 가장 중요한 샘플들 간의 거리계산법은 2절에서 제시한 $D_I$, $D_{KLD}$, $D_{FWD}$를 사용했다. 그리고 한 클러스터에 포함될 수 있는 기준을 두 샘플이 표현하는 한자의 획 수 차이가 20% 미만인 것으로 정의했다. 이는 두 샘플이 갖는 획의 수가 차이가 일정 이상 크다면 굳이 거리계산을 하지 않더라도 유사하지 않음을 알 수 있기 때문이다. 이와 유사한 개념으로 입력된 관측 심볼 열에 대해 인식을 할 때 관측 심볼 열의 길이와 비례관계에 있는 HMM의 상태 수를 고려하여 모든 HMM에 대해 출력확률을 구하는 것을 피한다.

클러스터링 과정은 $K$-평균 클러스터링($K$-Means Clustering)에 기초한다. 이때 사용하는 거리계산법은 샘플(HMM) 간의 상태 수가 같다는 하는 조건이 있으므로 같은 상태 수를 갖는 샘플끼리 분류하여 개별적으로 클러스터링을 진행한다. 본 논문에서 훈련한 샘플들의 상태 수는 2~70이므로 총 69개의 집합으로 분류된다. 분류된 샘플들에 적용하는 $K$ 값은 해당 샘플 집합의 크기를 상수 α=15로 나눈 값으로 적용한다. 이는 서로 다른 상태 수를 갖는 샘플 집합의 크기를 고려하여 가변적으로 적용한 것이다. 또한, 클러스터링 과정에서 한 클러스터가 너무 비대해지는 것을 막으려고 클러스터 분할과정을 포함한다. 클러스터 분할 방식은 어떤 클러스터의 샘플 수가 상수 α*2보다 크다면 해당 클러스터의 센터 값을 복사하여 두 클러스터로 분배되도록 유도하는 방식이다. 분할과정이 있으므로 최종적으로 구성된 클러스터의 수는 초기 $K$ 값과 다를 수 있다. 그림 2는 본 논문에서 적용한 클러스터링의 과정이다. 여기서 주의할 것은 샘플과 클러스터 센터의 거리를 계산할 때, 거리 값이 같은 센터가 2개 이상일 경우엔 포함된 샘플 수가 적은 센터의 클러스터를 선택해야 한다. 클러스터 센터 값 $\lambda_c$은 클러스터 $c$에 속한 모든 샘플 $m$이 갖는 파라미터 확률 값의 평균을 취한다. 물론 샘플의 파라미터 확률이외에 획의 수와 같은 다른 정보들도 모두 평균을 취한다.

### 4.2 인식과정

클러스터링 비기반의 인식시스템에서는 주어진 입력 관측 열 $O$에 대한 인식과정을 간단하게 설명할 수

```
Input:
  E = {e1,...,eN}        (Samples to be clustered)
  K = n(E)/α             (Number of clusters)
Outputs:
  C = {c1,...,cN}        (cluster centroids)
  m : E → {1,...,N}      (cluster membership)

Procedure KMeans
  Set C to initial value (e.g. random selection of E)
  While m has changed
    For each ci ∈ C
      Recompute ci as the centroid of {e|m(e) = i}
      If n(ci) > α*2 Then
        divide ci
      End
    End
    For each ei ∈ E
      For each ci ∈ C
        comment : Less than 20% of the difference in stroke number ei and ci is TRUE.
        If isValid(ei, ci) Then
          m(ei) = arg min distance(ei, cj)
                  j ∈ {1,...,N}
        End
      End
    End
  End
End
```

그림 2 수정된 K-평균 알고리즘

있다. 모든 HMM과 출력확률 계산을 통해 인식 결과를 사용자에게 알려주면 된다. 여기서 '모든 HMM'이라는 표현을 좀 더 엄밀하게 말하자면 입력 관측 열 $O$의 길이와 상태 수 $N$ 사이의 차이가 20% 내외인 HMM이 해당한다. 이는 본 논문에서 HMM을 훈련할 때 사용된 관측 열 길이의 평균으로 상태 수 $N$을 결정하기 때문이다.

마찬가지로 주어진 입력 관측 열 $O$에 대해 클러스터링 기반의 인식시스템에서의 인식과정은 그림 3과 같이 진행된다. 여기서 추가된 점은 먼저 클러스터 센터와의 출력확률을 계산한 후 그 확률을 기준으로 클러스터에 포함된 샘플(HMM)의 수가 $K$개가 넘지 않도록 클러스터를 선택한다. 선택된 $K$개의 HMM에 대해서 다시 출력확률 계산을 하고 최종적으로 인식 결과를 출력한다. 여기서 $K$값은 입력 관측 열의 길이에 가변적으로 적용하는 상수로서 실험을 통해 측정된 값이다. 표 1에서 제시한 $K$값은 적절한 수준의 인식률을 확보하기 위해 측정된 값임을 밝힌다. 클러스터링 기반의 인식 시스템에서는 인식을 하기 위해 전체 HMM과 출력확률을 구하는 대신에 $K$개의 HMM을 선택하기 때문에 클러스터링 비기반의 인식시스템보다 더 작은 탐색 범위를 갖는다. 이 때문에 인식에 걸리는 시간이 줄어든다. $K$는 인식률과 인식속도 간의 반비례관계를 만들어주는 값이다. $K$ 값을 축소할수록 탐색범위가 줄어들기 때문에 인식시간은 더 단축되지

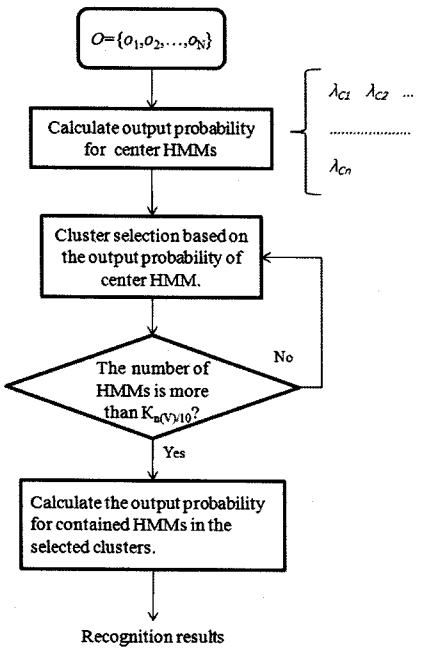


그림 3 클러스터링 기반 시스템에서의 인식과정

표 1 상수 $K$

| Range of observation symbol length | $K_n$ | Value |
|---|---|---|
| 2~9 | $K_0$ | 1000 |
| 10~19 | $K_1$ | 3000 |
| 20~39 | $K_2$ | 5000 |
| 40~49 | $K_3$ | 3000 |
| 50~ | $K_{4\sim}$ | 1000 |

만 인식률은 더 낮아질 수 있다. 이는 실험을 통해 알아보겠다.

## 5. 실험 및 결과분석

### 5.1 실험 환경 및 DB

본 논문에서 구현한 인식 시스템은 한·중·일 통합한자 u4e00~u9fa5까지 정의된 20,902 클래스를 인식 대상으로 한다. 이를 위해 남녀 44명이 쓴 27벌(20,902*27)의 샘플을 수집했다. 한자 샘플 수집에 참가한 남녀의 연령대는 20대에서 50대까지 다양하다. 수집된 샘플은 샘플 수집에 참여하지 않은 다른 사람을 통해 검수과정을 거쳤고, 이 과정을 통해서 수집된 데이터 중에 형태가 올바르지 못한 것은 사용하지 않았다.

총 27벌의 샘플 중에 34명이 쓴 22벌에 해당하는 564,354개의 샘플은 HMM을 훈련하는데 사용하고 나머지 10명이 쓴 104,510개는 인식률을 구하기 위한 테스트 샘플로 사용됐다. 수집된 필기 한자 샘플은 많은 환경적 요인에 영향을 받기 때문에 이를 보정하기 위해 튀는 점과 삐침 제거, 중복 샘플점 필터링 등의 전처리 과정[14]을 거친 후에 구조 코드(Structure code) 열로 변환해 관측 심볼 열로 사용했다. HMM의 구조는 문자 인식에서 널리 사용되는 왼쪽-오른쪽(left-to-right) 구조이고 Baum-Welch 알고리즘을 사용해 훈련했는데 이때 상태 수는 클래스마다 수집된 관측 심볼 열 길이의 평균으로 정하되 상태 수의 하한 값을 '2'로 상한 값을 '70'으로 했다(그림 4).

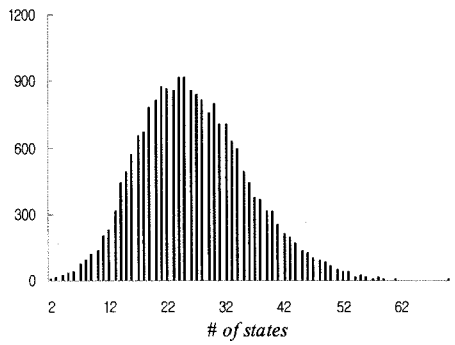


그림 4 상태 수로 분류한 HMM의 수

본 논문에서는 실험을 위해 거리계산법을 제외한 클러스터링 과정에서 사용되는 모든 설정은 같게 하고, 거리계산법만을 변경하여 각각의 거리계산법에 따라 개별적으로 진행했다. 4절 클러스터링 과정에 3 가지 거리계산법 $D_I$, $D_{KLD}$, $D_{FWD}$을 적용하여 이에 대응하는 3 가지 클러스터 모델 $M_I$, $M_{KLD}$, $M_{FWD}$을 생성했다. 생성된 클러스터 모델에서 클러스터의 개수는 $M_I$는 2053개 이고, $M_{KLD}$는 2122개, $M_{FWD}$는 2033개이다. $M_F$는 클러스터링을 하지 않은 시스템을 의미하고, 인식 대상이 되는 클래스의 수와 같은 20,902개의 모델을 갖는다. 클러스터 모델들이 서로 다른 클러스터 개수를 갖는 이유는 샘플들의 거리계산 값에 따라서 클러스터의 분할과정 빈도수가 달라지기 때문이다.

서로 다른 거리계산법을 적용한 $M_I$, $M_{KLD}$, $M_{FWD}$ 그리고 클러스터링을 하지 않은 $M_F$, 총 4개의 모델에 대한 비교 평가는 각각의 모델들을 인식시스템에 적용하여 테스트 샘플에 대한 인식률과 이때 인식에 걸리는 평균 시간을 측정하는 과정으로 이뤄진다. 실험에 사용된 PC는 인텔 CPU Core2Duo 3.0GHz, 메인 메모리 3.0GB, 윈도 2003 운영체제를 사용했다.

### 5.2 실험결과

그림 5는 클러스터링 기반 인식 시스템에서 줄어든 탐색 범위를 나타낸다. 여기서 범례 $M_C$는 클러스터링 모델 $M_I$, $M_{KLD}$, $M_{FWD}$을 의미한다. $M_C$를 적용한 인식 시스템에서는 클러스터 센터와의 출력확률을 이용해 클러스터에 포함된 샘플의 수가 $K$개를 넘지 않도록 클러스터를 선택해 탐색 범위를 축소했다. 이 때문에 클러스터링을 하지 않은 인식 시스템 $M_F$과 비교해 더 작은 탐색 범위를 갖고, 더 빠른 인식 속도를 갖는다. 서로 다른 클러스터 모델은 $K$ 값에 따라서 인식속도와 인식률에 영향을 받으므로 공정한 평가를 위해 클러스터 모델의 경우 같은 $K$ 값을 적용하였다.

클러스터링 기반 인식 시스템에서는 $M_F$와 비교해 클러스터 센터와의 출력확률을 구하는 과정과 클러스터 센터를 메모리에 적재하는 비용이 추가로 필요하다. 하지만, 클러스터 센터의 개수가 얼마 되지 않아서 추가로 요구되는 메모리가 20MB 정도밖에 안 되고 클러스터 센터와의 출력확률 계산을 통해 줄인 탐색범위는 추가로 필요한 클러스터 센터와의 출력확률 계산량보다 더 많은 계산량의 절감되므로 인식 시간을 단축하는데 큰 영향을 미치지 않았다.

그림 6, 7의 $X$축은 $K$ 값에 곱한 $\beta$ 값에 따라서 측정된 결과이다. 그림 6은 10순위 후보까지의 인식률을 보여주고 그림 7은 이에 대응하는 인식시간을 보여준다. 여기서 $\beta$=1.0일 때 표 1과 같은 범위를 갖는다. 범례 $M_F$에 해당하는 시스템은 클러스터링 기반의 시스템이 아니기 때문에 $\beta$ 값에 영향을 받지 않고 클러스터링 기반 시스템의 인식률 상한 값과 인식시간의 상한 값을 정해준다. 그림 7을 보면 서로 다른 거리계산법을 사용한 클러스터링 기반 시스템이 인식에 걸리는 시간이 거의 같은 것을 볼 수 있다. 이는 인식에 걸리는 시간은 상수 $K$ 값에 의존되는 것이지 거리계산법에 영향을 거의 받지 않기 때문이다.

클러스터링 기반의 시스템에서는 $K$가 작을수록 탐색범위가 줄어들어 인식에 걸리는 시간은 단축되지만, 인식률은 $M_F$에 비해서 낮아진다. 이는 정인식에 해당하는 HMM이 $K$개의 샘플을 선택하는 과정에서 빠질 확률이 점점 커지기 때문이다. 이러한 특성을 이용해 $K$ 값을 적절히 조절한다면 구현하려는 시스템의 목적에 맞게 유연하게 사용할 수 있다.

표 2는 그림 6과 그림 7의 결과를 정리한 것이다. 같은 $K$ 값에 대해서 $M_{FWD} > M_{KLD} > M_I$ 순서로 더 좋은 인식률을 확인할 수 있었다. 또한, 탐색 범위를 축소할수록 본 논문에서 제안한 거리계산법을 사

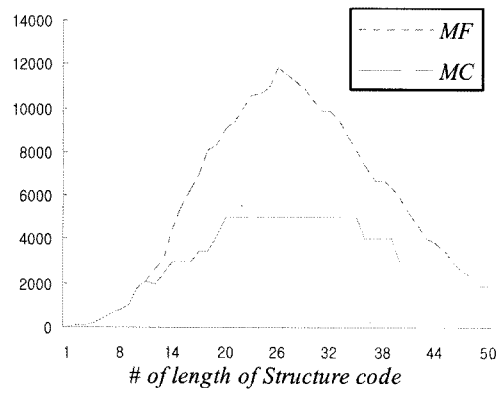


그림 5 구조코드열 길이에 따른 HMM 탐색 범위

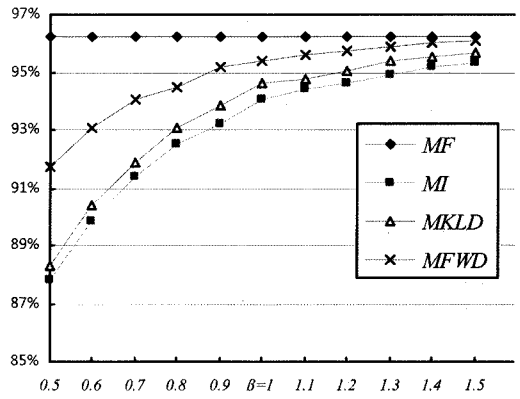


그림 6 인식률

표 2 $\beta$에 따른 인식률과 인식 소요 시간

| Model | $\beta$ | | | | | |
|---|---|---|---|---|---|---|
| | 0.5 | | $\beta$= 1.0 | | 1.5 | |
| | Rate | ms/char | Rate | ms/char | Rate | ms/char |
| $M_F$ | 96.26 | 512 | 96.26 | 512 | 96.26 | 512 |
| $M_I$ | 87.78 | 128 | 94.03 | 245 | 95.29 | 365 |
| $M_{KLD}$ | 88.29 | 135 | 94.62 | 245 | 95.69 | 362 |
| $M_{FWD}$ | 91.75 | 130 | 95.37 | 243 | 96.06 | 360 |

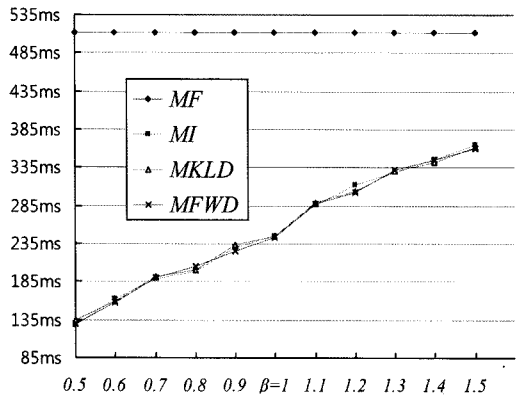


그림 7 인식에 걸리는 시간 ms/char

용한 $M_{FWD}$와 $M_{KLD}$가 기존 연구에서 제안한 거리계산법을 사용한 것 $M_I$보다 더 좋은 결과를 보였고, 가장 좋은 인식률을 보여준 것은 $M_{FWD}$였다. 이러한 결과는 $D_{FWD}$ 거리계산법이 더 자연스러운 클러스터 구성을 이끌어 냈음을 말해주고 HMM의 두 가지 파라미터만을 사용해 간단한 수식을 적용하였음에도 좋은 결과를 얻을 수 있음을 보여주었다. 하지만, 두 HMM 간의 출력확률을 비교함에 있어서 KLD를 적용하는 것은 좋은 결과를 얻지 못했다. $\beta$=1.0일 때 $M_{FWD}$의 경우에 $M_F$ 시스템이 평균 512ms의 인식 시간을 갖는 것에 비해 약 2배 향상된 평균 243m의 인식 속도를 보여주었고, 10순위 인식률은 $M_F$ 시스템의 96.26%에서 0.9%가량 낮아진 95.37%를 보여줬다. 마찬가지로 $M_{FWD}$를 적용한 인식 시스템에서 $\beta$=1.5일 때 인식률은 $M_F$ 시스템보다 0.2% 만이 낮아졌고, 인식 시간은 30% 정도 향상되어 평균 360ms의 인식속도 개선을 보여주었다.

## 6. 결 론

대용량 필기 한자 인식 시스템의 구현에서는 인식 대상인 클래스의 수가 방대하여 계산 자원 부족, 인식에 걸리는 시간의 증가 등의 다양한 문제점이 있다. 이 중 인식에 걸리는 시간의 증가는 좋은 시스템을 개발하는데 큰 걸림돌로 작용한다. 본 논문에서는 대용량 필기 한자 인식 시스템에서의 인식 시간 증가를 극복하기 위해서 $K$-평균 클러스터링을 적용하고, 이 때 필요한 거리계산법에 대해 제안했다.

유니코드 한·중·일 통합한자에 정의된 20,902자에 대한 온라인 필기 인식을 위해 본 논문에서 제안한 거리계산법을 사용한 클러스터링을 사용하여 인식시스템을 구현하였고, 실험결과 클러스터링을 하지 않았을 때와 비교해 약 2배의 인식 속도 향상을 가져왔고, 기존 연구를 통해 제안된 거리계산법에 비해 더 좋은 인식률을 보여줬다. 이러한 결과는 HMM의 중요한 두 가지 파라미터인 전이 확률과 관측 심볼 출력확률을 결합하여 은닉 상태 간의 전이를 잘 표현하도록 거리계산법에 적용한 결과라고 볼 수 있다.

인식 속도 측면에서는 개선은 만족할만한 수준이다. 하지만, 상수 $K$에 비례하여 인식률도 낮아지는 단점이 있다. 이러한 단점은 거리계산법의 개선 또는 클러스터링 알고리즘의 개선으로 보완할 수 있다. 이를 위해 적절한 상수 $K$값을 자동으로 찾는 방법, HMM 간의 거리계산에서의 상한 값을 정하는 방법 등과 HMM을 클러스터링하는 방법에 대한 더 많은 연구가 있어야 한다.

## 참 고 문 헌

[1] C. C. Tappert, C. Y. Suen, and T. Wakahara, "The State of the Art in On-Line Handwriting Recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol.12, no.8, pp.787-808, 1990.

[2] R. Nag, K.H. Wong, and F. Fallside, "Script Recognition Using Hidden Markov Models," *Proc. TCASSP '86*, vol.3, pp.2,071-2,074, 1986.

[3] S. Bercu, G. Lorette, "On-Line Handwritten Word Recognition: An Approach Based on Hidden Markov Models," *Proc. Third IWFHR*, pp.385-390, 1993.

[4] A. P. Dempster, N. M. Laird and D.B. Rubin, "Maximum Likelihood Incomplete Data via EM Algorithm," *Journal of the Royal Statistical Society*, Series B, vol.39, pp.1-38, 1977.

[5] H. Lucke, "Which Stochastic Models Allow Baum-Welch Training?," *IEEE Trans. Signal Processing*, vol.44, no.11, 1996.

[6] G. D. Forney, "The Viterbi algorithm," *Proc. of the IEEE*, 61:268-278, 1973.

[7] http://unicode.org/charts/PDF/U4E00.pdf, Unified CJK Ideographs.

[8] S. Kullback and R. A. Leibler, "On information and sufficiency," *Ann. Math. Statist.*, vol.22, pp. 79-86, 1951.

[9] J. Silva, S. Narayanan, "Average divergence dis-

tance as a statistical discrimination measure for hidden Markov models," *IEEE Trans. Audio, Speech, and Language Processing*, vol.14, no.3, pp.890-906, 2006.

[10] M. Falkhausen, H. Reininger, and D. Wolf, "Calculation of distance measures between Hidden Markov Models," *Forth European Conference on Speech Communication and Technology*, pp. 1487-1490, 1995.

[11] B. H. Juang and L. Rabier, "A probabilistic distance measure for hidden Markov models," *AT&T Technical Journal*, vol.64, no.2, pp. 391-408, 1985.

[12] M. Vihola, M. Harju, P. Salmela, J. Suontausta and J. Savela, "Two dissimilarity measures for HMMs and their application in phoneme model clustering," in *Proc. ICASSP 2002*, pp.933-936, 2002.

[13] M. N. Do, "Fast approximation of Kullback-Leibler distance for dependence trees and hidden Markov models," *IEEE Signal Proc. Lett.*, vol.10, no.4, pp.115-119, 2003.

[14] J. Y. Ha, "Structure code for HMM Network-based Hangul Recognition," 18th *International Conference on Computer Processing of Oriental Language*, pp.106-113, 1999.

김 광 섭

2007년 8월 강원대학교 컴퓨터정보통신공학전공(학사). 2009년 8월 강원대학교 컴퓨터정보통신공학과(석사). 관심분야는 패턴인식, 인공지능

하 진 영

1987년 2월 서울대학교 전자계산기공학과(학사). 1989년 2월 한국과학기술원 전산학과(석사). 1994년 2월 한국과학기술원 전산학과(박사). 1994년 3월~1997년 2월 (주)핸디소프트 기술연구소. 1997년 3월~현재 강원대학교 컴퓨터학부 교수. 2000년 7월~2001년 7월 IBM T.J. Watson Research Center 방문연구원. 관심분야는 패턴인식, HCI 등