論文93-30B-2-2

멀티 프로세서 시스템에 의한 고속 문자인식

(High Speed Character Recognition by Multiprocessor System)

崔 東 赫*, 柳 盛 元**, 崔 成 男*, 金 學 秀***, 李 鎔 均*, 朴 圭 泰*

(Dong Hyuk Choi, Seong Won Ryu, Sung Nam Choi, Hak Su Kim, Yong Gyun Lee, and Kyu Tae Park)

要 約

본 연구에서는 복합 문자체와 복합 크기 문자의 인식, 고속 처리 시스템을 개발하였다. 연구방향은 알고리듬의 단순성, 적응성, 학습 가능성, 계층적 정보처리, 피드 백에 의한 주의 처리이다. 복합크기 문자인식을 위해 정규화를 하였으며, 특징벡터를 추출하여 계층구조 분류기로 분류하였다. 특징 추출 방식은 방향성 에지(edge) 필터 처리후 방향성 수용장(receptive field)을 적용하여 추출하였다. 계층구조 분류기는 두단의 전분류, 한단의 최종 분류로 구성된다. 두단의 전분류 역할은 예측(prediction) 효과로 최종 분류시 전 탐색(full search) 대신 제한된 부분 탐색(partial search)으로 처리 속도 향상을 가져온다. 세가지 종류의 인쇄체 폰트로 구성된 세개의 문서, 총 1,700자에 대해 95%의 인식율을 얻었으며, 고속처리를 위하여 4개의 transputer로 구성된 링(ring) 구조 멀티 프로세서 시스템을 구현하여, 초당 30자의 인식 속도를 얻었다.

Abstract

A multi-font, multi-size and high speed character recognition system is designed. The design principles are simpilcity of algorithm, adaptibility, learnability, hierachical data processing and attention by feed back. For the multi-size character recognition, the extracted character images are normalized. A hierachical classifier classifies the feature vectors. Feature is extracted by applying the directional receptive field after the directional edge filter processing. The hierachical classifier is consist of two pre-classifiers and one decision making classifier. The effect of two pre-classifiers is prediction to the final decision making classifier. With the pre-classifiers, the time to compute the distance of the final classifier is reduced. Recognition rate is 95% for the three documents printed in three kinds of fonts, total 1,700 characters. For high speed implemention, a multiprocessor system with the ring structure of four transputers is implemented, and the recognition speed of 30 characters per second is aquired.

*正會員, **準會員, 延世大學校 電子工學科
(Dept. of Elec. Eng., Yonsei Univ.)
***正會員, 三星電子(株)
(Samsung Elec. Ltd.)
接受日字 : 1992年 3月 17日

I. 서론

고속 문자 인식을 위해서는 인식대상 문자, 알고리듬, 시스템 측면에서 고속처리를 위한 방법들이 면밀히 검토되어야하고, 이들 방법간 상호 연관에서 처리 속도를 저하시키지 않는 전반적 스케줄이 요구된다.

본 논문에서는 인식 대상 문자는 실제 사용 문서에 적합하도록 한글, 영문, 숫자, 특수기호로 범위를 삼았으며, 한자는 제외되었다. 문자인식을 고속으로 처리하기 위해 고려해야 할 항목으로 세 가지를 들 수 있다.

첫째, 한글과 같은 조합 문자의 경우 인식 시 자소 분리 수행 여부를 결정하여야 한다. 보통 기존의 자소 분리 알고리듬은 시간이 많이 소요되고 완벽하지도 못하다. 그러나 자소 분리를 수행하면 인식 대상 문자 수가 증가가 인식에 영향을 미치지 않는 장점이 있다. 자소 분리를 하지 않는 경우 자소 분리의 부담을 덜 수 있으나 인식 대상 문자가 증가할 경우 인식 혼동의 가능성이 증가한다. 본 논문에서는 고속처리를 위해 알고리듬의 단순성을 추구하였고 이에 따라 자소 분리를 하지 않고 인식하는 방식을 채택하였다. 인식 대상 문자 증가에 따른 인식율 저하는 한글에서 주로 사용되는 1000자를 인식대상으로 좁혀 해결하였으며, 이는 실제 문서 인식시 유용하리라 생각된다.

둘째, 특징 추출과 인식에서는 단순성과 고속처리를 위해 방향성 에지(edge) 추출과 방향성 수용장(receptive field)에 의한 벡터값을 특징으로 사용하였고 인식에는 패턴 분류 기법을 도입하였다. 논문 [1] 에서는 지도 학습에 의한 패턴 분류기를 형성하고 이에 의해 인식을 하였으나 인식 대상 문자 수가 증가함에 따라 인식 시간이 많이 소요되므로 논문 [2] 에서 제안된 계층구조 분류기를 채택하였고, 처리 시간을 보다 줄이고 구현된 시스템에 보다 적합하게 그 구조를 수정하였다.

셋째, 고속처리를 위한 시스템 구현에서는 알고리듬의 변경을 용이하게 하기 위해 특수 목적용 하드웨어 구현보다는 다수개의 transputer에 의한 병렬처리 시스템을 구현하였다. 본 논문의 문자인식 알고리듬은 특성상 파이프라인(pipeline)구조이며 개발된 시스템은 이에 적합하도록 transputer를 프로세서로 가지는 링구조의 멀티프로세서로 구현되었다.

본론에서는 문자인식 알고리듬을 소개하였으며, 실험 및 결과에서는 문자, 문서인식율과 시스템으로의 알고리듬 이식 및 처리과정을 기술하였다.

Ⅱ. 본론

1. 전체 처리과정

그림1에 전체 처리과정을 보였다. 스캐너 출력인 이진 영상에서 문자 추출후, 추출된 문자는 bilinear interpolation에 의해 32×32로 정규화 된다. 문자 추출 방법은 논문 [4] 에 있다. 특징 추출은 논문

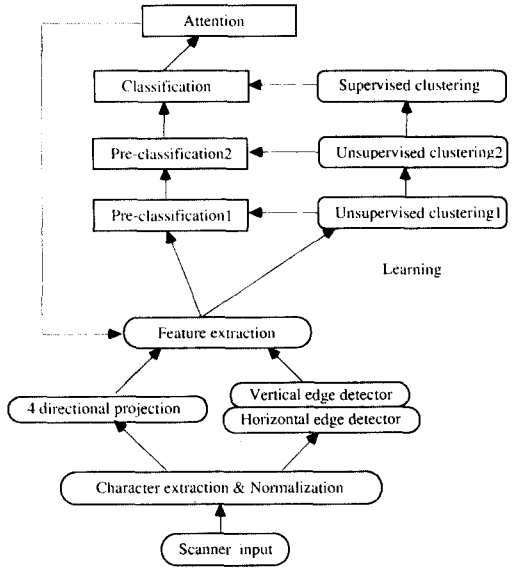


그림 1. 전체처리과정
Fig. 1. Overall processing.

[2] 에서 제시된 방향성 연산자를 적용하여, 그림1의 3단 분류기에 각각 입력된다. 각각의 특징을 정규화된 문자 F(x,y)에 대해 다음과 같이 추출하였다.

$$F_H(x,y)=\left|\sum_{j=-1}^{1}\sum_{i=-1}^{1}H(i,j)*F(x-i,y-j)\right| \quad (1)$$

$$F_V(x,y)=\left|\sum_{j=-1}^{1}\sum_{i=-1}^{1}V(i,j)*F(x-i,y-j)\right| \quad (2)$$

〈 16 크기 특징추출 〉

$$UDFH_{ij}=\sum_{x=8*i}^{8*i+8}\sum_{y=16*j}^{16*j+16}F_H(x,y)\ i=0,\cdots,4,\ j=0,\cdots 2 \quad (3)$$

$$UDFH_{ij}=\sum_{x=16*i}^{16*i+8}\sum_{y=8*j}^{8*j+16}F_V(x,y)\ i=0,\cdots,2,\ j=0,\cdots 4 \quad (4)$$

〈 32 크기 특징추출 〉

$$VDFH_{ij}=\sum_{x=8*i}^{8*i+8}\sum_{y=8*j}^{8*j+16}F_H(x,y)\ \ i=0,\cdots,4,\ j=0,\cdots 4 \quad (5)$$

$$VDFV_{ij}=\sum_{x=8*i}^{8*i+8}\sum_{y=8*j}^{8*j+8}F_V(x,y)\ \ i=0,\cdots,4,\ j=0,\cdots 4 \quad (6)$$

〈64 크기 특징추출〉

$$DFH_j = \sum_{x=4m}^{4m+4} \sum_{y=8n}^{8n+8} F_H(x,y) \quad i=0,\cdots,8, j=0,\cdots 4 \tag{7}$$

$$DFV_i = \sum_{x=8m}^{8m+8} \sum_{y=4n}^{4n+4} F_V(x,y) \quad i=0,\cdots,4, j=0,\cdots 8 \tag{8}$$

2. 학습과 분류

적응 계층 분류기는 두개의 비지도 학습 분류기(USACL1 : Unsupervised Adaptive Classifier1과 USACL2)와 하나의 지도 학습 분류기(SACL : Supervised Adaptive Classifier)로 구성된다. USACL1, 2는 입력노드와 출력노드를 가지고, 경쟁학습과 노드생성 임계치를 근거로 출력노드를 생성하는 비지도 분류기이다.[2] 이는 ART(Adaptive Resonance Theory)와 유사하나 ART의 top-down 매칭은 없다.[5] [11] SACL은 입력노드, 내부노드, 출력노드를 가지고 경쟁학습과 노드생성 임계치를 근거로 내부노드를 생성하는 지도 분류기이다.[1] [2] SACL 학습 방식은 LVQ(Learning Vector Quantizer)에 노드 생성 기법이 적용된 방식이다.[6] [9] SACL의 성능은 패턴 분류의 경우 LVQ, BP(Back-error Propagation)와 비슷하며 한글 문자 특징 분류시 보다 우수하다.[2] [10] 분류기 구조는 두 개의 비지도 분류기가 전분류에 해당되며, 최종단은 지도 학습 분류기가 되어 이들 3개의 분류기는 각 분류기의 출력 노드단을 학습에 의해 상호 연결하여 계층 구조를 가진다. (그림 2참조)

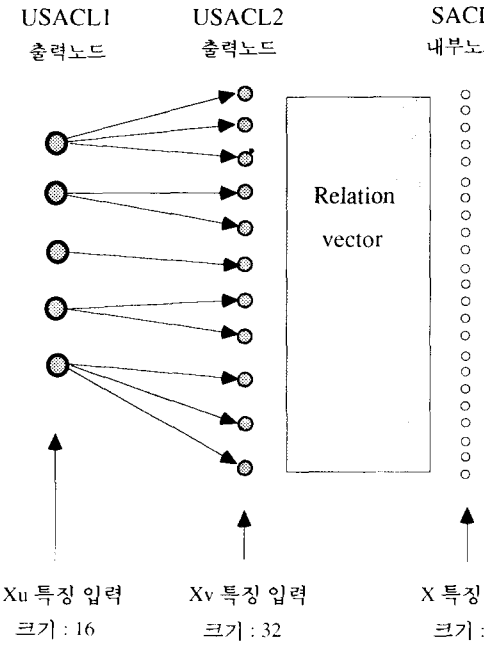


그림 2. 계층 분류기에서 각 분류기 출력노드, 내부노드의 상관관계

Fig. 2. The relation of the output node and inner node of each classifier.

이 장에서 사용되는 표현의 정의는 다음과 같다.

X_u : USACL1의 입력특징, UDFH, UDFV의 일차원 확장

X_v : USACL2의 입력특징, UVDFH, UVDFV의 일차원 확장

X : SACL의 입력특징, DFH, DFV의 일차원 확장

U : USACL1의 입력단과 출력단 간의 weight vector

V : USACL2의 입력단과 출력단 간의 weight vector

M : SACL의 입력단과 내부단 간의 weight vector

T_u : USACL1 노드확장의 기준값인 노드생성 임계치

T_v : USACL2 노드확장의 기준값인 노드생성 임계치

T_s : SACL 내부노드확장의 기준값인 노드생성 임계치

N_u : USACL1 출력 노드의 갯수

N_v : USACL2 출력 노드의 갯수

N_s : SACL 내부노드 갯수

C_{uk} : USACL1 노드 k에서 학습된 특징 벡터 갯수

C_{vk} : USACL2 노드 k에서 학습된 특징 벡터 갯수

C_{sk} : SACL 노드 k에서 학습된 특징 벡터 갯수

〈USACL의 학습〉

초기화 : Nu = Nv = 0

STEP 1. USACL1 입력 특징 벡터 Xu와 출력 노드들간 거리 계산.

$$Dist_j^2 = \sum_{i=0}^{16} (x_{ui} - u_{ji})^2 \quad j = 1,\cdots, Nu \tag{9}$$

STEP 2. 최소거리 출력 노드 선정.

$$Dist_k^2 = \min_j \{Dist_j^2\}, \quad j = 1,\cdots, Nu \tag{10}$$

STEP 3. 최소거리가 USACL1의 노드생성 임계치 T_u 보다 작으면 STEP 5로 가고, 아니면 STEP 4로 간다.

STEP 4. USCAL1 출력 노드를 생성, $N_u = N_u + 1$, 생성된 출력노드의 대표벡터 U_{Nu}를 현 입력으로 대치, $C_{uNu} = 1$, STEP 8로 간다.

$$Nu_u = X_u \tag{11}$$

STEP 5. U_k갱신, $C_{uk} = C_{uk} + 1$, STEP 6으로 간다.

$$\Delta U_k = \frac{1}{C_{uk}}(X_u - U_k) \tag{12}$$

STEP 6. 입력 특징벡터 X_v를 USACL2 입력단에 넣고 USACL1 현 출력 노드와 연결된 출력 노드들과 거리 계산.

$$Dist_j^2 = \sum_{i=0}^{32}(x_i - v_{ji})^2 \tag{13}$$

, 여기서 j는 USACL1 출력노드와 연결된 USACL2 출력노드

STEP 7. 최소 거리 출력 노드를 선택, 최소 거리가 USACL2 노드생성 임계치 T_v보다 크면 새로운 출력 노드를 만든 다음 STEP 8로 간다. 아니면 STEP 9로 간다.

STEP 8. 새로운 USACL2 출력노드 생성, $N_v = N_v + 1$, 생성된 출력 노드의 대표 벡터 V_{Nv}를 현 입력 패턴 벡터 X_v로 대치, $C_{vNv} = 1$, USACL1 출력노드와 연결. STEP 1로 간다.

STEP 9. V_k를 갱신, $C_{vk} = C_{vk} + 1$.

$$\Delta V_k = \frac{1}{C_{vk}}(X_v - V_k) \tag{14}$$

〈SACL 학습〉

초기화 : Ns = 0, 출력노드 갯수 = 학습시킬 클래스 갯수.

STEP 1. 입력 특징벡터 X를 입력단에 넣고 이 입력 패턴이 소속된 클래스에 해당하는 출력 노드를 지정한다. 지정된 출력 노드와 연결된 내부 노드들과 학습 패턴 벡터간 거리를 식 (15)로 계산한다. 연결된 내부노드가 없으면 STEP 3으로 간다.

$$Dist_j^2 = \sum_{i=0}^{64}(x_i - m_{ji})^2 \tag{15}$$

STEP 2. 최소 거리 내부 노드 k를 선택, 이 최소 거리가 노드생성 임계치 T_s 보다 크면, 새로운 내부 노드를 만든 다음 지정된 출력 노드와 연결하고 STEP 3으로 간다. T_s 보다 작으면 STEP 4로 간다.

STEP 3. 내부 노드 생성 후, $N_s = N_s + 1$, M_{Ns}을 현 입력 패턴으로 대치, 생성된 내부노드를 소속된 클래스에 해당하는 출력노드와 연결한다. $C_{sNs} = 1$, STEP 1로 간다.

$$Mn_s = X \tag{16}$$

STEP 4. M_k 갱신, $C_{sk} = C_{sk} + 1$, STEP 1로 간다.

$$\Delta M_k = \frac{1}{C_{sk}}(X - M_k) \tag{17}$$

〈 상관 벡터의 학습 〉

상관 벡터 R(i,j)는 USACL2출력노드 i와 SACL 내부노드 j간 상관관계를 학습한 결과이다.

STEP 1.

R(i,j) = 0 으로 초기화.

i 는 USACL2의 출력노드.

j 는 SACL 내부노드.

STEP 2

USACL2 최소거리 출력노드 i에 대해,

SACL 최소거리 내부노드가 j일때,

$R(i,j) = R(i,j) + 1$

〈분류〉

STEP 1. USACL1에 X_u 입력 후, 거리 계산. Uncertainty threshold를 만족하는 출력 노드 선정.

$Dist_j^2 / Dist_k^2 > Un\text{-}th, \quad j = 0, \cdots, N$

k는 최소거리 출력노드.

STEP 2. USACL2에 Xv 입력 후, 거리 계산. Uncertainty threshold를 만족하는 출력 노드 선정.

$Dist_j^2 / Dist_k^2 > Un\text{-}th,$

j는 선정된 USACL1 출력노드와 연결된 USACL2 출력노드

k는 최소거리 출력노드.

STEP 3. SACL에 X 입력 후, USACL2 최소거리 출력노드 i에 대해 $R(i,j) > 0$ 을 만족하는 SACL 내부노드 M_j와 X 간 거리 계산. 거리를 기준으로 두 개의 후보를 결정.

3. 주의 처리(Attention processing)

주의(attention) 처리를 하는 주된 이유는 입력 패턴의 특정 부분만을 고려함으로써 분류시 발생하는 애매성을 최소화 하는 것이다. SACL 내부노드중 최소거리 2개의 후보로부터 주의에 의해 판정한다. 주의 처리는 예로 '을'과 '올'의 경우 두 글자의 유사성으로 해서 분류시 애매성이 생기며, 이는 모음 'ㅗ'만을 주의영역으로 고려하여 재결정를 시도하는 것이다. 본 논문에서는 영문, 숫자, 특수기호, 한글 자음은 제외하였고, 비교적 주의 영역결정이 쉽고, 처리가 용이하도록 모음 영역만을 고려하여 다음과 같이 처리하였다.

STEP1.
SACL 내부노드 두 개에 대해
$Dist_j^2 / Dist_k^2 > 0.7$ 이면, STEP 2로 간다.
아니면, 최소거리로 판정.
j는 최소 다음거리 SACL 내부노드.
k는 최소거리 SACL 내부노드,

STEP2.
후보 두개에 대해 모음코드만이 틀리면,
STEP3.
아니면, 최소거리 판정.

STEP3.
모음영역의 선정.
구조특징추출.
2개 후보중 최종 결정
STEP1로 간다.

모음 영역의 선정은 SACL단에서 추출되어 나온 두개의 후보 문자코드를 초성, 중성, 종성으로 분해하여 모음의 모양을 알아낸 다음 미리 준비된 표에서 영역을 추출한다. 모음의 종류는 받침의 있고 없음, 수직방향, 수평방향, 자음과 조합될 때의 모양등을 고려하여 미리 13가지의 형을 부여한다. 각 형은 주의영역을 가지게 된다.

다음 선정된 모음 영역에 대해 구조 특징을 추출한다. 한글 모음은 수직, 수평으로 구성되어 있으므로 그림 3과 같은 4방향 projection 벡터를 구한다.

그림 3과 같이 projection에 대해 th=5보다 크면, d [i] 값은 1이 된다. 이와 같이 추출된 특징 벡터 d에 대해 그림 4와 같은 모음 구조 특징을 추출한다.

주의 영역에서 다루는 모음 대상은 단순 모음으로 그림 4에서 추출한 A, B, C, D로 한글 모음을 결정한다.

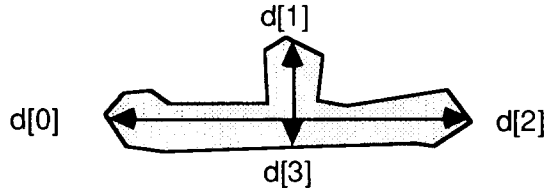


그림 3. 4방향 porjection 특징 백터
Fig. 3. Four directional projection feature vector.

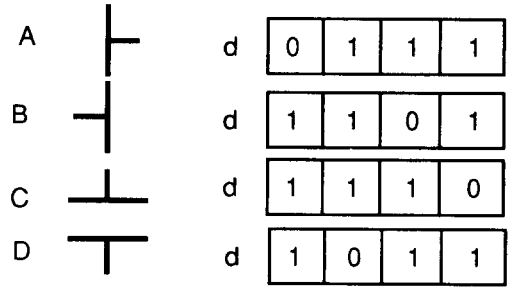


그림 4. 구조특징 추출
Fig. 4. Structure extraction.

Ⅲ. 실험 및 결과

1. 개별문자실험

개별 문자 실험 데이타는 multi-font, multi-size 44종류로 구성되어 있으며, 23종류의 한글, 영문, 숫자, 특수기호 8,320자로 학습하고, 21종류의 한글, 영문, 숫자, 특수기호 13,167자의 데이타로 실험하여 92% 이상의 인식율을 얻었다. [3] 각 폰트 속성은 논문 [3] 에 있다.

2. 고속처리

멀티 프로세싱에는 프로세서간 데이타 이동 방식으로 공통 버스(commom bus) 방식과 메세지 전달(message passing) 방식 두 가지가 있다. 공통 버스는 버스 충돌을 회피하기 위한 버스 제어가 필요하며, 메세지 전달 방식은 프로세서간 통신을 위한 링크(link)와 프로토콜이 필요하다. transputer는 링크를 통한 메세지 전달 방식을 사용하고 있다. 링크를 통한 통신은 점-대-점(pont-to-point)통신이며, CPU 실행과 병행하여 통신이 이루어진다. 링크를 통한 확장으로 scalable performane 획득이 가능하다. 링크의 데이타 전달 속도는 10Mbits/sec이다. [13] [14]

본 연구에서는 transputer 4개로 링크를 이용하여

링 구조로 시스템을 구현하였다. 시스템 구성은 그림 5와 같다. 이중 root transputer는 링 구조를 위해 Slave1, Slave3와 연결해야 하고, 또한 IBM-PC와도 연결해야 하므로 3개의 링크가 필요하다. 따라서 T400과 성능은 유사하면서 링크가 4개인 T425를 사용하였다. slave transputer는 두 개의 링크만이 필요하므로 링크가 두 개인 T400을 사용하였다. [13]

다음 개발된 알고리듬을 링 구조 멀티 프로세서에 이식시키고 처리되는 방법을 살펴본다. 전체 처리과정은 크게 정규화, 특징추출, 전분류, 분류, 주의처리 5개의 task로 구성되며, 처리전 학습된 데이터를 필요로 하는 프로세서에 전송시키는 초기화 작업이 필요하며 그림 6에 보였다. 각 task는 그림 7과 같이 프로세서에 할당되어 파이프라인으로 처리된다. 처리시간으로 초당 30자의 처리속도를 얻었다.

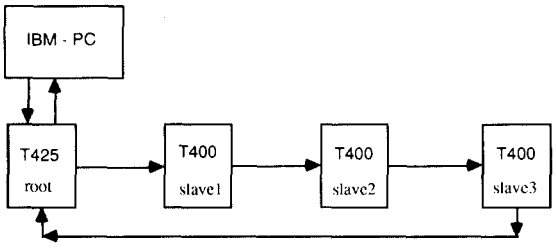


그림 5. 멀티 프로세서 시스템 구조
Fig. 5. The structure of multiprocessing system.

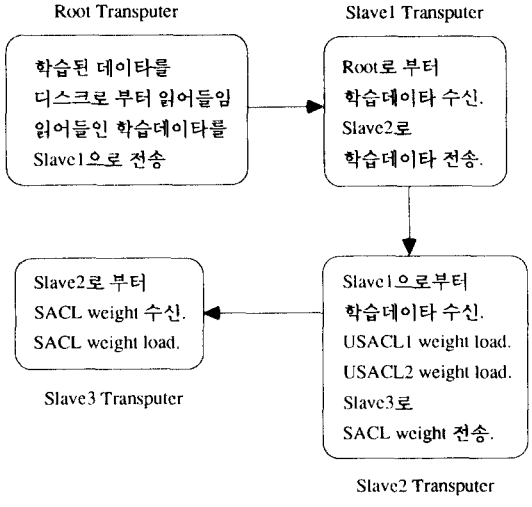


그림 6. 각 프로세서의 초기화
Fig. 6. Initialization of each processor.

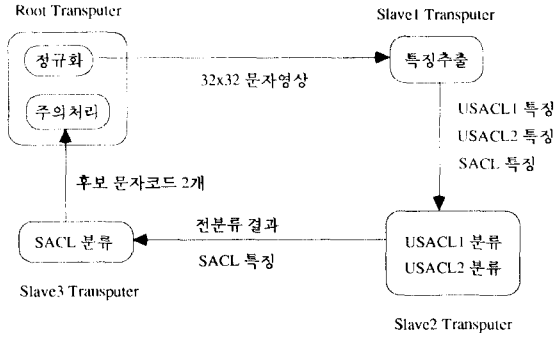


그림 7. 각 프로세서 간 전송데이타
Fig. 7. Transmission data between each processor.

3. 문서 인식

문서 인식 실험을 위해 실제 사용되는 문자체를 사용하였으며, 사용된 문자체는 세가지로 아래한글 명조체 레이저 프린터 출력, SuperWrite 명조체 레이저 프린트 출력, SuperWrite 고딕체 출력이다. 문서에서의 문자 추출은 논문 [4] 의 방법으로 추출하였다. 문자 추출시 생기는 문제점으로 한글 2형식에서 '가' 글자의 경우 'ㄱ' 과 'ㅏ'로 분리되어 추출되므로 학습시 한글 자음, 모음을 부가하여 학습하였으며, 인식후 결합하였다. 또한 문서인식에 사용되는 학습데이타는 세가지 문서의 문자 폰트로 학습하였으며, 각 폰트당 학습시킨 문자수는 한글 사용 빈도순 1000자, 영문자, 숫자, 특수기호이다. 학습 결과 USACL1의 노드수는 106개, USACL2의 노드수는 1153개, SACL의 노드수는 2882개가 발생하였다.

문서 인식시 USACL1, USACL2의 Uncertainty threshold값을 0.7로 하였으며, 세가지 문서 인식 결과중 각 부분을 그림 8, 9, 10에 보였으며, 인식율은 표 1에 보였다. 처리시간은 초당 30자이며, 대표적인 오인식 문자의 예를 표 2에 보였다.

표 1. 문서 인식율
Table 1. The text recognition rate.

문서종류	문서내 문자갯수	인식율
아래한글 명조체 문서(그림 8의 (a))	765개	96.1%
SuperWrite 명조체 문서(그림 9의 (a))	720개	95.3%
SuperWrite 고딕체 문서(그림 10의 (a))	215개	95.3%

표 2. 오인식 문자의 예
Table 2. The examples of misrecognized characters.

(a) 아래한글 명조체 문서

입력문자	위 델 H 추 로 위 o 론 링 를 며 는 로 교 처
인식결과	워 텔 ㅐ 주 르 취 으 른 렁 틀 벼 논 토 고 저

(b) SuperWrite 명조체 문서

입력문자	본 합 층 H 즘 위 함 적 류 분 e 류 습 을 증 였 템 u
인식결과	분 항 총 R 츰 워 항 격 유 운 6 루 승 울 즘 었 탬 n

(c) SuperWrite 고딕체 문서

입력문자	우 를 과 습 L
인식결과	두 틀 파 슴 ㄴ

오인식 결과를 보면

a) 자음 중, 추〈-〉주, 를〈-〉틀, 함〈-〉합〈-〉항, 과〈-〉파 들이 구분이 잘 안되는데 그 이유는 특징 추출 방법이 에지의 크기를 문자내 블럭마다 합해서 벡터 형식으로 이루어지므로 자음의 세부 차이가 잘 표현되지 못하기 때문이다. 또한 자음의 경우 주의 처리가 없기 때문이다.

b) 모음 중, ㅗ 〈-〉 ㅡ 〈-〉 ㅜ, ㅓ 〈-〉 ㅣ 〈-〉 ㅏ, ㅓ 〈-〉 ㅕ, ㅜ 〈-〉 ㅠ에서 모음의 경우에는 주의 처리를 하였음에도 오류가 생기는 원인은 첫째, 분류결과 두 개의 후보를 가지고 처리를 하게 되는데, 두 개의 후보중 올바른 후보가 없는 경우이다. 둘째는 모음 구조 특징 추출이 완벽하지 못하기 때문이다.

c) o〈-〉 0, I 〈-〉 l 〈-〉 ㅣ, L 〈-〉 ㄴ의 혼동은 입력 패턴이 거의 유사하기 때문에 문맥에 의한 처리가 요구된다.

알고리즘의 단순성은 특징 추출시 주 알고리즘은 mask에 의한 correlation을 사용하고, 패턴간 유사도 기준을 correlation이나 유클리디언 거리(Euclidian Distance)를 사용하므로써 얻어지며, 위와 같은 유사도를 사용하기 위해 패턴으로 부터의 모든 특징은 일정한 크기를 가지는 벡터로 표현된다. 다음 병렬 처리성, 학습 가능성은 통계적 분류방법 혹은 신경망(Neural Network)에 의한 분류 방식을 채택함으로서 가능하다. 그러나 적응성, 계층적 정보 처리는 통계적 방법으

그림 8. (a)아래한글 명조체 문서
Fig. 8. (a)The Munjo font text of the HWP(Hangeul Word Processor).

그림 8. (b)아래한글 명조체 문서의 문자 영상
Fig. 8. (b)The character image of the Munjo font text of the HWP.

알고리 즘의 단순성은 특징 주출시 주 알고리즘은 mask에 의한 c○ rre lat ion을 사용하고, 패턴간 유사도 기준을 corre lat ion이나 유클리디 언 거 리 (Euc l id ian Di stance)를 사용하므르써 얻어지 며, 위와 같은 유사도를 사용하기 취해 패턴으로 부터의 모든 특징은 일정한 크기를 가지는 벡터토 표현된다. 다음 병렬 처리성, 학습 가능성은 통계적 분류방법 혹은 신경망 (Neural Netw으rk)에 의한 분류 방식을 채택함으로서 가능하다. 그러나 적응성, 계층적 정보 처 리는 통계적 방법으

그림 8. (c) 아래한글 명조체 문서 인식 결과.
Fig. 8. (c) The recognition result of the Munjo font text of the HWP.

요 약

본 연구에서의 문자인식 연구대상은 복합 문자체와 복합 크기 문자의 인식, 고속처리이다. 연구방향은 알고리즘의 단순성, 병렬 처리성, 적응성, 학습 가능성, 계층적 정보처리, feed back에 의한 주의 집중, 그리고 H/W구성의 용어성이다.

복합크기 문자인식을 위해 입력되는 모든 문자는 32×32로 정규화된다. 추출되는 특징은 알고리즘의 단순성을 위해 백터로 표현된다. 특징을 백터 형식으로 추출함으로서, 이후 인식 방법은 통계적 분류방법과 신경망 분류방

그림 9. (a) SuperWrite 명조체 문서.
Fig. 9. (a) The Munjo font text of the SuperWrite.

그림 9. (b) SuperWrite 명조체 문서의 문자 영상.
Fig. 9. (b) The character image of the Munjo font text of the SuperWrite.

요 약

분 연구에서의 문자인식 연구대상은 복합 문자체와 복항 크기 문자의 인식, 고속처리 이다. 연구방향은 알고리즘의 단순성, 병렬 처리성, 적응성, 학습 가능성, 계층적 정보처리, feed back에 의한 주의 집중, 그리고 R/w구성의 용어성 이다.

복합크기 문자인식을 위해 입력되는 모든 문자는 32x 32로 정규화된다. 추출되는 특징은 알고리츰의 단순성을 워해 백터로 표현된다. 특징을 백터 형식으로 추출항으로서, 이후 인식 방법은 통계걱 분유방법과 신경망 운류방

그림 9. (c) SuperWite 명조체 문서 인식 결과
Fig. 9. (c) The recognition result of the Munjo font text of the SuperWrite.

클래스는 하나 또는 몇 개만 내부 노드를 갖게되며, 넓은 영역에 분포된 패턴벡터를 갖는 클래스는 많은 내부 노드를 갖게 된다.

학습시에는 각 노드 별로 갖고 있는 대표 벡타 M을 입력 패턴에 따라서 수정해 나가며, 새로운 노드가 필요할 경우에 노드를 만들어 주고 이 노드와 출력 노드를 연결 해 주는 일을 하게 된다.

〈학습 순서〉

STEP 1. 입력 패턴X를 입력단에 넣고 이 입력패턴이 소속된 클래스에 해당하는 출력 노드를 지정한다. 지정된 출력 노드와 연결된 내부 노드들과 학습 패턴 벡터간 거리를 식(4.11)로 계산한다.

그림 10. (a) SuperWite 고딕체 문서.
Fig. 10. (a) The Gothic font text of the SuperWrite.

그림 10. (b) SuperWite 고딕체 문서의 문자 영상
Fig. 10. (b) The character image of the Gothic font text of the SuperWrite.

클래스는 하나 ㅍ는 몇 개만 L 내부 노드를 갖게 되며. 넓은 영역에
분포된 패턴벡터를 갖는 클래스는 많은 L 내부 노드를 갖게 된다.
학습시 에는 각 노드 별로 갖고 있는 대표 벡타 M을 입력 패턴에
따라서 수정 해 나 가 며, 새로운 노드 가 필요할 경두에 노드를 만들어 주고
이 노드와 출력 노드를 연결 해 주는 일을 하게 된다.
<학습 순서>
STE P I. 입력 패턴X를 입력단에 넣고 이 입력패턴이 소속된 클래스에
해당하는 출력 노드를 지 정한다. 지 정된 출력 노드아 연결된 내부 노드들파
학습 패턴 벡터간 거리를 식 (4111)로 계산한다.

그림 10. (c) SuperWrite 고딕체 Laser 출력문서 인식결과.
Fig. 10. (c) The recognition result of the Gothic font text of the SuperWrite.

Ⅳ. 결론

본 연구에서는 복합 문자체와 복합 크기 문자의 인식 , 고속 처리 시스템을 개발하였다. 연구방향은 알고리듬의 단순성, 적응성, 학습 가능성, 계층적 정보처리, feed back에 의한 주의 집중이다. 문자 추출을 제외한 문서 인식 실험 결과 초당 30자의 고속처리로 실현 가능성을 보였다.

본 연구의 문자 인식에서 특징 추출 방식은 방향성 에지 필터처리, 방향성 수용장에 대한 에지값의 합이다. 분류기는 패턴간 유사도를 유클리디언 거리로 삼고, 경쟁학습, 적응적 노드확장 학습, 최소거리 결정이 주요 사항이다. 이러한 접근방향의 특성은 실험결과 다음과 같이 요약된다.

1) 분류기 최종 결정은 사용된 특징이 에지의 분포 특징이므로 정확한 결정을 위한 주의처리가 요구된다.

2) 문자 인식 결과는 학습된 데이타에 의존한다. 즉 학습 알고리듬에 의한 일반성 획득과 폰트에 무관한 특징 추출은 어려운 문제이다.

3) 알고리듬이 단순하므로 병렬처리가 용이하며 고속처리가 가능하다.

4) 학습데이타 저장을 위한 메모리 요구가 크다.

앞으로의 연구방향은 주의처리의 확장 및 강화, 처리속도의 향상으로 실용성을 높이는 것이다.

參 考 文 獻

[1] 최 원호, 최 동혁, 이 병래, 박 규태, "한글인식을 위한 신경망 분류기의 응용," 대한 전자공학회 논문지, 제27권 제8호, 1990, pp. 93-103.

[2] 최 동혁, 강 현철, 류 성원, 박 규태, "계층구조 신경망을 이용한 한글 인식," 대한 전자공학회 논문집, 제28권, B편 제11호, 1991, pp. 1-7.

[3] 최 동혁, 류 성원, 김 학수, 이 병래, 박 규태, "계층구조 신경망을 이용한 문자 인식," 신호처리 합동학술대회, 1991. 9.

[4] 김 학수, 류 성원, 최 동혁, 박 규태, "신경망을 이용한 문서분류에 관한 연구,"대한 전자공학회 추계 학술대회, 1991. 11

[5] S. Grossberg, Neural Networks and Naturall Intelligence, MIT press, 1988.

[6] T. Kohonen, Self-organization and Associative Memory, Springer-Verlag, 1984.

[7] Yoh-Han Pao, Adaptive Pattern Recognition and Neural Networks, Addison-Wesley, 1989.

[8] P. K. Simpson, Artificial Neural System, Pergammon press, 1990.

[9] T. Kohonen, "An introduction to neural computing," Neural Networks, vol. 1, pp. 3-16, 1988.

[10] T. Kohonen, R. Chrisley, and G. Barma, "Statistical pattern recognition with neural networks : benchmarking studies," Neural Networks from Models

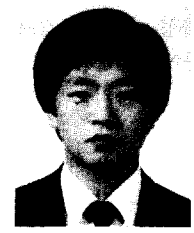
to Applications, I.D.S.E.T., Paris, pp. 160-167, 1989.
[11] R. O. Lippmann, "An introduction to computing with Neural Nets," IEEE ASSP Magazine, pp. 4-22, April 1987.
[12] R. O. Duda and P. E. Hart, Pattern Classification and Scene Analysis, Wiley-Interscience, 1973.
[13] "The Transputer Application Notebook," INMOS, 1989.
[14] "The Transputer Development and iq. Sysytem Data Book," SGS-THOMSOM, 1989

著 者 紹 介

崔 東 赫 (正會員)
1963年 5月 7日生. 1986年 연세대학교 전자공학과 졸업. 1988年 연세대학교 전자공학과 석사 졸업. 1988年 - 현재 연세대학교 전자공학과 박사과정. 주관심 분야는 패턴 인식, 영상처리 등임.

金 學 秀 (正會員)
1985年 연세대학교 전자공학과 졸업. 1985-90年 삼성전자 정보통신 연구소 근무. 1992年 연세대학교 전자공학과 석사 졸업. 1992年 - 현재 삼성전자 정보통신 연구소 요소연구실 주임 연구원. 주관심 분야는 영상처리, 컴퓨터 네트워크 등임.

柳 盛 元 (準會員) 第 28 卷 B編 第 11 號 參照
현재 연세대학교 전자공학과 박사과정.



李 鎔 均 (正會員)
1953年 8月 19日生. 1977年 연세대학교 전자공학과 졸업. 1980年 연세대학교 전자공학과 석사 졸업. 1989年 - 현재 연세대학교 전자공학과 박사과정. 1980年 - 현재 한국전자통신 연구소 ISDN 신호처리 연구실장 주관심 분야는 신호처리, 교환시스템, 소프트웨어 공학 등임.

崔 成 男 (正會員) 第 27 卷 第 3 號 參照
현재 연세대학교 전자공학과 박사과정.

朴 圭 泰 (正會員) 第 28 卷 B編 第 11 號 參照
현재 연세대학교 전자공학과 교수.