한글 문서 인식 시스템 SILNOON*

이 승호, 조 창재, 김 일영, 오 세창, 조 성배, 김 진형
한국과학기술원 전산학과

SILNOON: A Hangul Document Recognition System

Seungho Lee, Changje Cho, Ilyoung Kim, Sechang Oh, Seongbae Cho and Jin H. Kim
Department of Computer Science, KAIST

요 약

본 논문에서는 한국과학기술원 전산학과 인공지능연구실에서 개발하고 있는 한글 문서 인식 시스템 SILNOON을 소개한다. 본 연구는 인쇄체 한글로 작성된 문서를 자동으로 인식하여 컴퓨터 화일로 저장하고, 인식된 문서를 편집 및 수정하여 레이저 프린터를 통하여 출력할 수 있는 실용적인 한글 문서 인식 시스템의 개발을 그 목적으로 하고 있다. SILNOON 시스템은 크게 전처리, 문자 인식, 후처리 등의 세 단계로 구성되어 있다. 본 논문에서는 SILNOON 시스템의 각 구성 단계에 대하여 설명하고 개인용 컴퓨터 상에서 구현되어 있는 시제품을 가지고서 실험한 결과를 발표한다.

I. 서 론

최근에는 전자 기술의 발달로 종이를 사용하지 않고 의사를 전달할 수 있는 많은 장치들이 발명되었다. 방대한 양의 자료를 적은 공간을 사용하여 저장할 수 있고 효율적으로 필요한 자료를 찾을 수 있도록 하는 컴퓨터가 바로 그러한 것의 하나이다. 그러나 이의 적극적인 활용을 위해서는 기존의 많은 방대한 양의 기록들이 컴퓨터에 미리 입력되어야만 하는데, 터미널의 키보드를 두드리거나 입력카드에 천공하는 등의 기존의 방법으로 자료를 입력할 경우에는 엄청난 인력과 시간이 소요된다. 그러므로 컴퓨터에 의한 정보화 사회의 구현을 위해서는 반드시 기계에 의한 자동 입력 시스템이 필요하다. 즉, 사람이 단지 컴퓨터에 문서를 보여줌으로써 그 문서를 입력할 수 있는 시스템이 개발되어야 한다.

본 논문에서는 한국과학기술원 전산학과 인공지능연구실에서 개발한 한글 문서 인식 시스템 *SILNOON*을 소개한다. 한글 문서 인식 시스템은 우리가 일상적으로 사용하는 한글 문서의 인식을 목표로 한다. 여기서의 한글 문서란 잡지, 신문, 연구보고서 및 서류 등을 일컫는다. 문서는 도형, 사진과 같이 비문자 영역과 본문 및 기사 부분과 같은 문자 영역으로 구분된다. 이 시스템은 문서의 영상을 입력 장치로부터 받아들여 도표 및 사진 부분은 컴퓨터에서 처리할 수 있는 영상 자료로, 문자 부분은 각 문자에 해당되는 부호로 출력할 수 있도록 설계되어 있다.

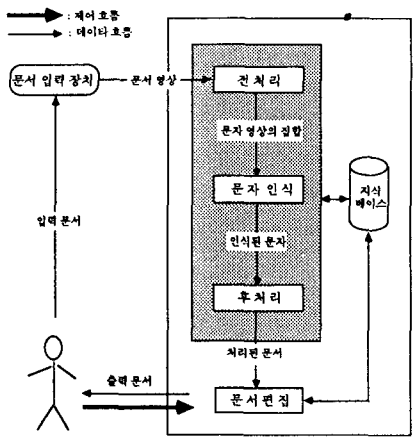
II. 한글 문서 인식 시스템의 개요

한글 문서인식 시스템은 그림 1과 같이 크게 전처리, 문자 인식, 후처리의 세 단계로 구성되어 있다[6].

전처리 단계는 입력 문서 영상을 그림 부분과 문자 부분으로 구분한 다음 그림 부분은 영상자료 형태로, 문자 부분은 문자 영상의 집합으로 보관하여 문자 인식과정에서 처리할 수 있도록 해 주는 과정이다. 이 과정에서는 문서 영상이 항상 똑바로 들어온다고 가정할 수 없기 때문에 입력된 문서 영상의 기울어진 각도를 계산하고 그 각도만큼 영상을 돌려서 바로 잡아야 한다. 그리고 또한 입력 문서의 특성에 따라서 전체적인 구조를 분석하여야 한다. 문서 영상을 분석하여 문자 영역과 비문자 영역을 분리하고 문자 영역에 대해서는 각 문자 영상을 추출하여 문자 인식 단계에 넘겨 주고, 비문자 영역은 영상 자료의 형태로 저장하게 된다.

문자 인식 단계는 전처리 과정로부터 넘겨 받은 문자 단위의 영상을 분석하여 이를 부호화하는 단계이다. 이 단계에서는 문자 영상으로부터 특성을 추출한 다음, 그 특성에 의하여 문자를 인식한다. 본 시스템에서는 문자의 형태가 고정된 인쇄체 문자를 대상으로 하였기 때문에 mesh 특성이라는 비교적 단순한 통계적 특성에 의하여 문자를 인식하였다.

문서 인식 시스템의 마지막 단계는 후처리 단계이다. 이 단계에서는 문자 단위 영상만을 독립적으로 분석하여 인식된 결과를 확대하여 전체를 볼 수 있는 관점에서 오류를 수정하는 과

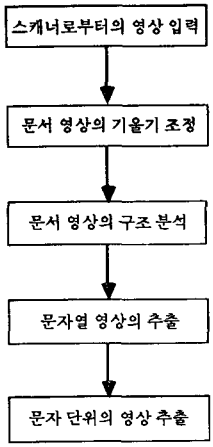


< 그림 1 : SILNOON 시스템의 구성 >

정이다. 예를 들어 문자 인식 단계에서 "컴퓨터"라는 인식 결과가 나왔다면 이는 필히 "컴퓨터"의 오인식일 것이다. 아무리 좋은 인식 알고리즘을 사용하더라도 입력 문서에 포함된 잡음과 입력 문자들의 유사성으로 인하여 문자를 항상 정확하게 인식할 수 없기 때문에 오인식된 문자를 문맥에 의하여 수정해 줄 수 있는 확인 단계가 있어야만 신뢰성 있는 문서 인식 시스템을 구성할 수 있다. 이 단계에서는 사용된 인식 알고리즘의 특성, 즉 어느 글자가 어느 글자로 잘못 인식되는 경우의 수가 많았다는 등의 통계적 자료와 음절의 출현빈도, 혹은 단어의 출현 가능성 여부 등의 여러가지 정보를 종합하여 인식된 문자들을 어절 단위로 수정한다.

III. 전처리 단계

전처리 단계는 입력 문서 영상을 비문자 부분과 문자 부분으로 구분한 다음 비문자 부분은 영상자료 형태로, 문자 부분은 문자 영상의 집합으로 보관하여 문자 인식과정에서 처리할 수 있도록 해 주는 과정이다.



< 그림 2 : 전처리 단계의 구성 >

모든 형식의 문서에 적용 가능한 전처리 알고리즘의 개발은 매우 어려운 문제이므로 특정한 문서의 형식에 대한 경험적 지식을 많이 사용하고 있다. 따라서 특정 응용 분야가 지정되면 보다 많은 경험적 지식이 채택 가능하므로 문서구조 분석과정에서의 정확성을 향상시킬 수 있다.

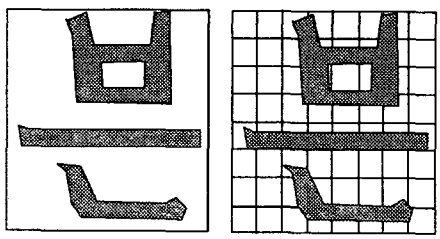
SILNOON 시스템에서 사용한 전처리 시스템은 그림 2와 같이 5 개의 과정으로 구성되어 있다[6]. 이러한 과정들을 거쳐서 문서 영상으로부터 개개의 문자 영상을 추출하여 문자인식 단계로 넘겨주게 된다.

IV. 문자 인식 단계

문자를 인식하는 방법으로는 원형비교 방법, 통계적 방법 및 구문론적 방법등이 있다. 문서 인식 시스템에서의 인쇄체 문자에 대한 인식 알고리즘은 기본적인 연산이 간단하여 하드웨어로 쉽게 구현되고 인식율도 높아야 하는데 통계적 방법은 이러한 특성을 잘 반영한다.

4.1 특성 추출

SILNOON 시스템에서는 통계적 방법으로 한글을 인식하기 위하여 문자의 전체적인 특성을 반영하는 mesh 특성점을 사용하여 각 문자의 특성을 추출한다[9]. 이 특성점을 추출하는 방법은 그림 3과 같이 문자 패턴을 8x8의 부분지역(cell)으로 나눈 다음, 각 지역의 검은 부분의 면적에 해당하는 값을 계산하여 그 부분지역의 특성값으로 정한다. 따라서 mesh 특성점은 64개의 스칼라 값으로 구성된다.



< 그림 3 : mesh 특성 >

4.2 인식 알고리즘

통계적 방법으로 문자를 인식하는 과정은 먼저 입력 문자에 대해 특성점 벡터 X를 구한 다음, 그것과 시스템에 저장되어 있는 각 모델 문자의 특성 벡터와의 거리를 비교하여 그중에서 가장 가까운 문자 모델로 분류하는 방법이다. 입력 문자의 특성 벡터 X는 64-차원의 벡터, $X = (x_1,\ldots,x_{64})^t$로 표현된다. 각 문자의 특성 벡터의 값은 스캐너를 통하여 입력될 때마다 그 값이 조금씩 변하므로 여러 번 같은 문자에 대하여 그 값을 구하여 평균과 분산을 구할 필요가 있다. $M_i = (m_1, m_2,\ldots, m_{64})^t$를 i 번째 모델 문자의 특성점의 평균 벡터라 하고, V_i를 M_i의 각 특성값들 사이의 공분산을 표현하는 벡터라고 할 때, 입력 문자와 i 번째 모델 문자간의 유사성을 계산하는 식은 다음과 같다.

$$\min_i^{-1} (X - M_i)^t V_i^{-1} (X - M_i)$$

위의 식을 계산하는 데는 많은 실수 연산을 요하므로 일반적으로 다음과 같은 식으로 근사시켜서 계산을 하게 되는데, 여기서

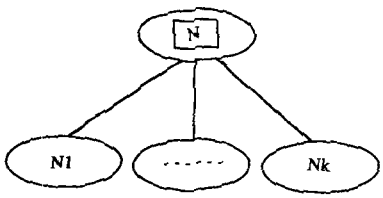
m_i은 M_i를 정수로 근사시킨 벡터이다.

$$\min_i^{-1} (X - m_j)^t (X - m_j)$$

단순한 선형 탐색에 의하여 입력 패턴에 가장 가까운 모델을 찾아 내는 방법은 매우 많은 연산 시간을 필요로 하며, 또한 시스템에 저장되어 있는 모델 문자의 수에 비례하여 계산시간이 증가한다는 단점을 가지고 있다. 그래서 본 시스템에서는 문자의 인식 시간 및 인식률을 향상시키기 위한 보다 개선된 방법으로 tree classifier를 구성하여 문자를 인식하였다.

4.3 Tree Classifier의 구성

각 모델은 64 차원의 벡터로 표현되는데 이 특성점들의 값의 크기에 의하여 clustering이 이루어지게 된다. 하나의 cluster는 tree에서 하나의 node로 표현되고 초기에 tree의 root node는 990자의 한글 모델 전체를 포함한다. 상위 cluster로부터 분리된 하위 cluster는 상위 cluster를 parent로 하는 child node가 된다. 각각의 child node는 다시 clustering에 의하여 여러 개의 cluster로 나누어지게 되는데, 이러한 일련의 과정은 각 node에 포함된 모델의 수가 어느 일정한 수보다 작아질 때까지 계속된다. 표 1은 Tree Classifier의 구성 방법을 간략히 기술한 것이다.



1. Select the largest FD's (8)
2. Clustering by k-means algorithm (k = 2)
3. Overlapping (N %)
   minimize | d(C1,Xi) - d(C2,Xi) | for all Xi's
4. Until (# of classes in a child node) < T,
   repeat 1 - 2 for each child node.

< 표 1 : Tree Classifier의 구성 >

가. 특성점의 선택

clustering을 하는데 있어서 64 개의 특성중에서 몇 개의 주요한 특성만을 선택하여 clustering할 때 사용한다. 주요한 특성이란 한 node를 clustering하기 위해서 문자들을 잘 분리해 줄 수 있고 잡음의 영향이 적은 특성을 말하는데, 이러한 특성은 Fisher가 정의한 기준[11]을 사용하여 선택하였으며 그 기준은 다음과 같다.

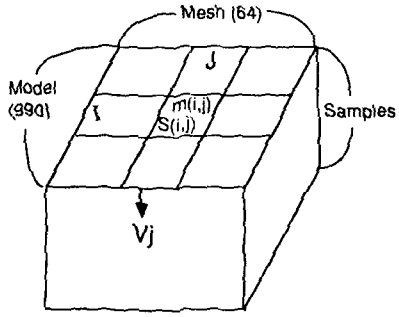
$FD_j = V(j) / (\Sigma_i S(i,j) / N),\ j = 0 \sim 63$

$V(j)$ = 전체 모델들의 j 번째 mesh 특성점의 분산

$S(i,j)$ = i 번째 모델의 j 번째 mesh 특성점의 분산

N = node에 포함된 모델의 갯수

SILNOON 시스템에서는 Tree Classifier를 구성하는 각 단계에서 64개의 특성값중에서 가장 큰 FD 값을 갖는 8개의 mesh 특성점을 선택하여 사용하였다.



Fisher's Discriminant Ratio (FD)

$$FDj = \frac{Vj}{(\Sigma Sij)/N}$$

< 그림 4 : Fisher의 Discriminant Ratio >

나. Clustering

앞에서 선택한 특성을 통해 각 모델은 8차원 벡터로 나타내어 질 수 있으며 이들 모델들은 k-means 알고리즘에 의하여 k개의 cluster로 분리한다[11]. 실험적으로 k의 값은 2내지 3이 적당한데 본 알고리즘에서는 이를 2로 하여 clustering을 하였으며, 이 값은 parameter로 주어질 수 있다. k-means 알고리즘에서는 초기의 cluster 중심이 주어져야 하는데, Local Minima에 빠지지 않게 하기위하여 우선 node에 포함된 모델들의 mesh의 평균을 구하고 이 값으로부터 거리가 가장 큰 모델을 첫번째 중심으로 선택하고 다시 이 중심으로 부터 거리가 가장 큰 모델을 또 하나의 중심으로 선택하여 Clustering을 수행하였다.

다. 중복(overlapping)

입력 문자에는 많은 잡음이 포함되어 있기 때문에 이러한 잡음으로 인해 tree의 탐색이 잘못된 방향으로 이루어질 수 있다. 따라서 이러한 문제점을 어느 정도 해결하기 위해서 중복하게 된다. node가 2 개의 cluster로 분리된 후 그 node에 포함된 각 모델들을 2 cluster의 중심과의 거리차가 작은 순으로 배열하고 이들중 그 node에 포함되어 있는 모델들의 수의 n%에 해당되는 상위 모델을 양 cluster에 모두 포함되도록 하는 것이다. 여기서 n도 parameter로 주어는데 여기서는 20%로 하여 Tree를 구성하였다.

라. 분류

입력 문자는 tree의 root로 부터 시작하여 그로부터 분리된 각 cluster의 중심과의 거리를 구하여 그 중에서 거리가 가장 작은 쪽으로 탐색을 진행한다. 이렇게 하여 terminal node에 이르게 되면 그에 포함된 모델 각각에 대해서 입력 문자의 특성 벡터와 거리를 계산하여 그 중에서 최소의 거리를 갖는 모델을 인식된 문자로서 결정한다.

V. 후처리 단계

문서 인식의 세번째 단계는 인식 확인 단계, 즉 오인식을 수정하는 단계이다. 이 단계에서는 사용한 인식 알고리즘의 특성, 즉 어느 글자가 어느 글자로 잘못 인식되는 경우의 수가 많았다는 등의 통계적 자료와 음절의 출현빈도, 혹은 단어의 출현 가능성 여부 등의 여러가지 정보를 종합하여야 한다. 사람은 단어내 문자의 연결관계, 단어간의 연결관계, 문장의 구조, 문서의 주제 등의 다양한 문맥적 지식을 이용하여 비교적 정확하게 문서 인식을 수행한다. 하지만 컴퓨터가 사람이 사용하는 모든 문맥적 지식으로 의미를 이해하고 수정하는 것은 구현상의 어려움과 많은 계산 비용으로 인하여 그 효율성이 줄어든다. 따라서 일반적으로 자동 문서인식 시스템에서는 단어내 문자의 연결관계에 관한 정보만을 사전으로부터 얻은 다음 그것을 이용하여 단어별로 오인식 수정을 하고 있다.

5.1 사전의 구성

한글 문서의 오인식 수정을 위해서는 효율적인 문맥적 지식을 제공할 수 있도록 사전을 구성하여야 한다. 이때, 사전의 구조는 이를 검색하는 알고리즘에 큰 영향을 미쳐서 오인식 수정 알고리즘의 효율성에 많은 영향을 줌으로 매우 중요하다.

SILNOON 시스템에서 사용된 사전은 어미나 조사 등의 활용이 발달되어 있는 한국어의 특성을 고려하여 각 어절들을 실사, 접사(affix), 허사등 세가지 부류의 사전으로 나누어 구성하였다. 실사 사전에는 체언, 용언의 어간, 부사 및 수식어 등을, 접사 사전에는 접미사 및 보조 어간을, 그리고 허사 사전에는 조사나 어미 등을 각각 저장하였다. 이때 실사와 허사는 끝음절부터 시작하여 음절의 역순으로 사전에 저장하였는데, 허사는 어절로 부터 조사 및 어미의 분리를 용이하게 하기 위하여, 또 실사는 복합어를 처리하기 위한 것이다. 실사 사전과 허사 사전에서는 음절의 많은 부분이 서로 같으므로 기억 공간을 줄이기 위하여 trie 구조로 사전을 구성하였다. 특히, 실사 사전을 trie 구조로 구성함으로써 복합어를 따로 저장할 필요가 없게 되어 기억 공간의 낭비를 줄일 수 있다. 이러한 방식으로 사전을 구성함으로써 사전의 크기와 탐색 시간을 줄일 수 있으며 효율적으로 사전을 관리할 수 있다.

이 외에도 오인식 수정 알고리즘에서 사용되는 사전으로 혼동확률(confusion probability) 사전과 사전확률(priori probability) 사전이 있다. 혼동확률 사전에는 어떤 문자가 문자인식 시스템에 의하여 어느 문자로 인식되느냐에 대한 정보를 저장되고 있고, 사전확률 사전에는 기계화 연구소에서 조사한 한글 잦기 순위중에서 상위 990 자의 한글에 대한 상대적인 빈도수를 저장하고 있다 [10].

5.2 오인식 수정 알고리즘

문자인식 시스템에서의 오인식 수정 알고리즘은 인식된 단어를 야기시킬 수 있는 단어중에서 확률이 가장 높은 것을 입력 단어로 결정한다. *SILNOON* 시스템에서 구현된 오인식 수정을 위한 후처리 알고리즘은 하향식 방법인 Dictionary Look-Up 알고리즘과 상향식 방법인 Modified Viterbi 알고리즘을 한글 문장의 띄어쓰기 단위인 어절의 특성에 맞게 결합시켜 개발되었다[8]. 이 알고리즘을 개략적으로 기술하면 다음과 같다.

문자인식 시스템에 입력된 입력어절을 $Z = Z_1Z_2...Z_n$이라 하고, 그 입력어절에 대한 인식어절을 $X = X_1X_2...X_n$이라 하자. 그러면 인식어절이 X일 때 입력어절이 Z일 확률 P(Z|X)는 Bayes의 정리에 의하여 다음과 같이 표현된다.

$$P(Z|X) = P(X|Z) * P(Z) / P(X) \text{ ------ (1)}$$

위의 (1) 식에서 P(X|Z)는 어절 Z가 입력되어 X로 인식될 어절간의 혼동확률(confusion probability)을 나타내며 문자인식 시스템의 특성을 반영한다. P(Z)와 P(X)는 각각 X와 Z의 사전확률(priori probability)을 나타낸다. 이와 같이 인식어절로 X가 주어졌을 때 모든 입력 가능한 어절중에서 (1) 식에서의 P(Z|X)를 가장 크게하는 어절 Z를 구하는 것이 오인식 수정 알고리즘의 목적이다. (1) 식에서 우변의 분모항인 P(X)는 가능한 모든 Z에 대하여 공통이고 또 Z와 X는 독립이므로, P(Z|X)를 가장 크게 만드는 입력 단어는 아래의 (2) 식의 값을 가장 크게 만드는 Z가 된다.

$$G(X,Z) = P(X|Z) * P(Z) \text{ ------ (2)}$$

이때, 계산의 편의를 위하여 (1)' 식의 우변에 Log 함수를 취한 것을 G(X,Z)으로 정하면 다음과 같이 된다.

$$G(X,Z) = \log P(X|Z) + \log P(Z) \text{ ------ (3)}$$

인쇄체 문자의 인식 시스템에서는 그 특성상 한 문자와 그 다음에 인식될 문자 사이에 조건부 독립(conditional independence)이 성립하고, 하나의 인식문자와 전체 입력문자 사이에는 Markov 가정이 성립되므로 어절간의 혼동확률은 각 문자들의 혼동확률의 곱으로 나타낼 수 있다.

$$\begin{aligned} P(X|Z) &= P(X_1X_2...X_n|Z_1Z_2...Z_n) \\ &= P(X_1|Z_1Z_2...Z_n) \ ... \ P(X_n|Z_1Z_2...Z_n) \\ &= P(X_1|Z_1) \ P(X_2|Z_2) \ ... \ P(X_n|Z_n) \text{ ------ (4)} \end{aligned}$$

그리고 P(Z) 즉, 어절들의 사전확률은 일반적으로 구하기가 매우 어려우므로 어휘 사전과 어절 형성 문법을 이용하여 어절을 이루는 각 토큰의 사전확률의 곱으로 어절의 사전 확률을 근사시켰다[8]. 예를 들어 어절 "교사는"이 나올 확률 P(교사는)은 그 어절을 이루고 있는 토큰 "교사"와 "는"의 사전 확률의 곱인 P(교사) * P(는)으로 근사시킨다. 그리고 각 토큰의 사전확률은 어휘 사전의 탐색 정보와 토큰을 구성하고 있는 각 음절의 사전확률의 곱으로써 구한다. n개의 문자로 이루어진 어절 Z가 어절 형성 문법을 만족하는 m개의 토큰 T_i로 나누어지고 결합 가능한 토큰간의 조건부 독립이 성립한다고 가정하면 P(Z)은 다음과 같이 근사시킬 수 있다

$$\begin{aligned} P(Z) &= P(T_1 \ ... \ T_m) \\ &= P(T_m) \ P(T_{m-1}|T_m) \ ... \ P(T_1|T_2 \ ... \ T_m) \\ &= P(T_m) \ ... \ P(T_1) \text{ ------ (5)} \end{aligned}$$

$$P(T_i) = f(T_i) + \Sigma_k P(D_{i,k})$$

$f(T_i)$ = 1000000 if T_i exists in Dictionary, 0 otherwise

$P(D_{i,k})$ = 토큰 T_i의 k 번째 음절의 사전확률

(4) 식과 (5) 식에 의하여 우리는 인식어절 X에 대하여 G(X,Z)를 가장 크게하는 어절 Z를 구할 수 있게 된다. 따라서 구하고자 하는 Z는 어절 형성 문법을 만족하는 Z 중에서 다음 식의 값을 가장 크게하는 Z가 된다.

$$G(X,Z) = \Sigma_{1\le i\le n} \log P(X_i|Z_i) + \Sigma_{1\le i\le m} \log P(T_i) \text{ --- (6)}$$

인식 어절	"고사은"					
혼동 확률	logP(고시교)	logP(고시고)	logP(사사)	logP(사시)	logP(은는)	logP(은은)
	-2.4	-1.2	-0.8	-2.8	-3.1	-0.1
사전 확률	logP(교사)	logP(교시)	logP(고사)	logP(고시)	logP(는)	logP(은)
	-5.1	-9.5	-5.4	-6.4	-3.2	-3.1
G(X,Z) 값	G(X,교사는)	G(X,교시는)	G(X,고사는)	G(X,고시는)		
	-14.6	-21.0	-13.7	-16.7		
수정 어절	"고사는"					

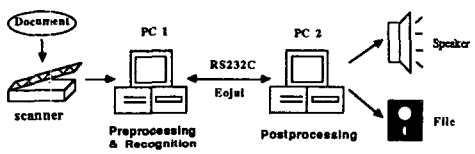
< 그림 5 : 후처리의 예 >

VI. 실험 및 결과 분석

SILNOON 시스템은 개인용 컴퓨터(IBM PC/386)상에 C 언어로 구현하였다. 실험한 한글 문자 집합은 기계화 연구소에서 발표한 한글 문자의 잦기 순서[10]에 의해 상위 1000 자의 문자중에서 생소한 문자들은 제외하고 외국어 표기를 위한 문자들을 추가해서 만든 990 자로 하였다. 이 990 자에 대한 누적 사용 빈도율은 99.0%이다.

SILNOON 시스템은 연구 보고서, 중학교 교과서, 가로쓰기 신문 등의 여러가지 형태의 한글 문서에 대하여 실험되었다. 그림 6은 SILNOON 데모 시스템의 구성을 보여 주고 있다.

표 2는 990 자의 문자를 포함하고 있는 10 장의 연구 보고서 양식의 문서에 대하여 문자인식과 후처리 과정을 실험해서 나온 결과를 보여 준다. 이 문서에 포함된 문자로 Qnix 레이저 빔 프린터의 H4 폰트를 사용하였다. 각 문서의 평균 어절 인식률과 평균 어절 수정률은 각각 97.17%와 98.08%로서 비교적 높은 인식률과 수정률을 나타내었으며, 이것을 문자단위로 지산하면 99.0%의 문자 인식률과 99.5%의 문자 수정률을 나타낸다. SILNOON 시스템이 1000 자가 포함되어 있는 문서를 처리하는 데 전처리, 문자인식 및 후처리 과정등을 모두 포함하여 약 3분 정도가 소요된다.



< 그림 6 : SILNOON 데모 시스템의 구성 >

실험 회임	어절수	글자수	문자 인식률 (%)	어절 인식률 (%)	어절 수정률 (%)
1	118	360	99.7	99.2	100.0
2	103	345	97.9	93.8	95.5
3	92	269	99.2	97.8	97.8
4	71	195	97.9	94.2	98.5
5	82	230	99.5	98.7	100.0
6	102	301	98.2	94.7	95.7
7	80	215	99.5	98.7	98.7
8	123	356	99.7	99.2	99.2
9	87	251	98.4	95.4	95.4
10	109	336	100.0	100.0	100.0

< 표 2 : 실험 결과 >

VII. 결 론

본 논문에서는 한국과학기술원 인공지능연구실에서 개발한 한글 문서 인식 시스템 SILNOON을 소개하였다. 이 시스템은 전처리, 문자 인식, 후처리 등의 세 단계로 구성되어 있다.

전처리 단계에서는 가로쓰기 전단으로 구성된 입력 문서에 대한 구조 분석 알고리즘을 사용하였다. 본 연구에서는 신문 기사의 구조를 분석하는 알고리즘[7]과 신문에 나오는 주식 시세 도표를 분석하는 알고리즘도 개발하였다.

문자 인식 알고리즘은 실제로 사용하는 크기의 문자의 인식에 중점을 두었지만, 인식은 특정 활자체에만 제한하였다. 그러나 SILNOON 시스템에서 제공하는 훈련 과정을 통하여 새로운 단일 문자체에도 쉽게 적응이 가능하다. 본 시스템이 실용화 되기 위해서는 여러 가지 종류와 크기의 다양한 문자체로 작성된 문서, 즉 한 문서에 여러 가지 종류의 문자체가 포함되어 있는 경우에 대한 연구도 수행되어야 하겠다.

입력 문자의 유사성과 문서 영상에 포함된 잡음으로 인하여 입력 문자를 항상 정확히 인식하지 못하기 때문에 실용적인 문서 인식 시스템을 만들기 위해서는 후처리 과정이 반드시 필요하다. 본 시스템에서 개발한 후처리 알고리즘을 문서 인식 시스템에 적용해 본 결과, 실험으로써 그 유용성이 증명되었다. 앞으로는 단순한 어휘적 지식뿐만이 아니라 구문적 지식과 의미적 지식 등을 이용하여 오인식을 수정할 수 있는 후처리 알고리즘에 관한 연구가 있어야 하겠다.

참고 문헌

[1] 이 주근, 이 광우, "한글 문자의 인식에 관한 연구(II)," 전자공학회지, 제 7권 제 3호, pp. 130-136, 1970년 12월.
[2] 김 태균, T. Agui, "Syntactic법에 의한 한글의 패턴 인식에 관한 연구," 전자공학회지, 제 14권 제 5호, pp. 154-160, 1977년 12월.
[3] 최 병욱, T. Ichikawa, H. Fujita, "한글 인식에 있어서의 자소추출," 전자공학회지, 제 18권 제 2호, pp. 36-43, 1981년 4월.
[4] 이 주근, 남궁 재찬, 김 영건, "한글 Pattern에서 Subpattern분리와 인식에 관한 연구," 전자공학회지, 제 18권 제 3호, pp. 1-8, 1981년 6월.
[5] 박 종욱, 이 주근, "Shape Pattern에 의한 필기체 한글 인식," 전자공학회지, 제 22권 제 5호, pp. 420-428, 1985년 9월.
[6] 이 성환, 강 희중, 김 형훈, 박 진규, 심 원태, 이 승호, 김 진형, "문서 인식 및 검색을 위한 전처리 시스템의 설계 및 구현," 한국정보과학회 추계 학술발표회 논문집, pp. 503-509, 1986년 10월.
[7] 김 형훈, 이 성환, 김 진형, "신문의 구조적 분석을 통한 한국 신문 기사의 추출," 한국정보과학회 학술논문지 제 15권 5 호, 1988년 10월, pp. 392-404.
[8] 박 진규, 김 진형, "한글 문서인식의 오인식 수정에 관한 연구," 한국정보과학회 추계 학술발표회 논문집, pp. 94-97, 1987년 10월.
[9] 이 성환, 조 창제, 김 진형, "실용적 한글 문서 자동 인식 시스템 개발의 문제점 및 개선 방향," 한국정보과학회 춘계 학술발표회 논문집, pp. 127-130, 1988년 4월.
[10] 한국 기계화 연구, 한글 기계화 연구소, 1975.
[11] J. T. Tou and R. C. Conzalez, *Pattern Recognition Principles*, Addison-Wesley, Massachusetts, 1974.

*본 연구는 과기처 특정 연구과제(기업주도형:
(주) 삼보 컴퓨터, 1987.4-1989.4)로 수행되었음