구조적 정보를 이용한 온라인 필기 한자 인식 결과 검증

윤병훈° 하진영
강원대학교 컴퓨터정보통신공학과
{ysoftman°, jyha}@kangwon.ac.kr

Result Verification of On-line Handwritten Chinese Character Recognition using Structural Information

Byoung-Hoon Yoon° Jin-Young Ha
Department of Computer and Information Communications Engineering, Kangwon National University

요 약

본 논문에서는 온라인 필기 한자 인식 과정에 있어 비슷한 한자끼리 혼동되어 오인식되는 한자들을 선별하여 이들 한자 인식과정의 후처리에 적용 할 수 있는 결과검증 방법에 대해서 소개한다. 결과검증 방법은 온라인 필기 한자인식기가 최종 인식결과를 산출하기 전에 후보한자들 중 혼동한자가 있으면 그 혼동한자에 대해서 미리 정해놓은 조건을 만족하는지 검사하여 점수를 내고 이를 인식 후처리에 이용한다. 인식 후처리에 쓰이는 결과검증 방법에서는 그 한자만이 가지고 있는 구조적인 정보를 다른 한자들과 구별하기 위해 휴리스틱으로 파악하여 조건화 시킨다. 구조적인 정보는 획의 좌표, 방향코드, 순서, 길이 등으로 판단되며 다양한 휴리스틱방법이 고려 될 수 있다. 인식 후처리에 적용되는 결과검증 방법을 통해 혼동되는 온라인 필기 한자를 구별하는데 도움이 되는 것을 실험을 통해 확인하였다.

1. 서 론

현재 온라인 필기 인식은 PDA와 같은 모바일 환경에서 중요한 입력장치로 자리 잡고 있다. 사람의 자연스러운 필기는 PDA와 같은 모바일기기에는 필수가 되었지만 필기자로부터 입력된 한자가 인식되지 않아 같은 한자를 몇 번이고 반복해서 입력해야 하는 불편한 경우가 종종 발생한다. 이런 한자들 중에는 사람의 눈으로는 쉽게 알 수 있는 한자들이 다수 포함되어 있다.

한자는 대부분 단순한 직선 모양을 가지는 획의 조합으로 구성되어 있다. 그러나 한자 자체의 방대한 문자 집합과 각 문자에 대한 획순과 획수의 다양성과 동일 문자에 약자, 속자가 존재하는 등 필기자 개개인의 다양한 필기 습관에 따라 여러 가지 문자 변형이 존재하게 된다[1].

보통 온라인 필기 한자 인식기는 인식기가 가지고 있는 모델 데이터와 필기자로부터 입력된 데이터 사이에 유사성을 판단하여 인식하게 되는데 사람은 쉽게 구별할 수 있는 한자를 인식기가 다른 한자로 오인식하는 경우가 발생한다. 이는 인식기의 모델 데이터가 충분하지 않아 입력 데이터와의 차이가 커서 상대적으로 비슷하면서 차이가 적은 다른 모델 데이터에 매칭이 되는 경우가 많다. 이렇게 오인식되는 경우 인식기의 모델 데이터를 추가하지 않고 한자의 구조적 정보를 휴리스틱(Heuristic)으로 파악하고 이를 조건화시켜 검사하는 결과검증(Result Verification) 방법을 이용하면 오인식을 줄이는데 도움이 된다.

결과검증 방법 실험을 위해 인식기가 쉽게 혼동하여 오인식되는 한자들은 따로 분류하여 결과검증 한자 대상으로 삼았다. 전체 인식대상 한자(2,361자)들 중 2획부터 10획 사이의 한자들로 한자의 모양이 서로 비슷하여 쉽게 혼동 될 수 있는 한자(226자)를 선별하였다. 결과검증 대상 한자들은 전체 인식 대상 한자의 약 10%에 해당하며 한자의 구조적 정보를 휴리스틱 방법으로 조건화 시켰다.

본 논문에서 사용한 온라인 필기 한자 인식기는 최종 인식결과를 산출하기 전에 중간인식결과로 10개 이하의 후보한자를 선별한다. 이 후보 한자들 중 혼동되어 오인식될 소지가 있는 한자들이 있으면 결과검증 방법을 사용하여 그 한자의 조건에 맞는지 검사하고 조건에 맞으면 보너스(Bonus) 점수를 주고 아니면 패널티(Penalty) 점수를 주어 최종 인식결과에 반영시킨다. 결과검증 방법을 온라인 필기 한자 인식기의 후처리에 적용시킴으로써 최종인식 결과를 바꿀 수 있도록 한다.

2. 혼동한자들의 구별 방법

필기 한자는 한자 자체의 방대한 문자 집합과 각 문자에 대한 획순과 획수의 다양성으로 인해 필기자 개개인의 다양한 필기 습관에 따라 여러 가지 문자 변형이 존재하게 된다. 이와 같은 변형의 다양성은 온라인 필기 한자 인식분야에 어려움을 가중시키고 있다[2].

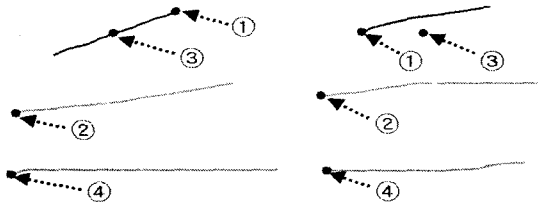
한자를 구성하는 기본단위는 획으로 정의할 수 있다. 획은 입력 디바이스에서 펜을 띄지 않고 한 번에 쓴 궤적을 의미한다[3]. 하나의 획에서 특징 파라미터를 추출하여 유사한 다른 획들과 비교할 수 있다.

2.1 2차원 좌표를 이용한 구별

온라인 필기 한자 인식에서 획은 시간에 따라 연속적으로 입력되는 점들로 이루어진다. 이 점들을 윈도우즈 좌

표 시스템(Windows Coordinate System)으로 표현하면 0이상의 양수 값을 가지는 x와 y로, 하나의 점을 (x,y)의 2차원 좌표로 나타낼 수 있다. 이 점들 중에서 처음 찍힌 시작점과 획의 끝나는 끝점까지 순서대로 입력된 점을 알 수가 있다. 그리고 한 획을 구성하는 점들 중에서 x최대/최소, y최대/최소(xmin, xmax, ymin, ymax, ...)값과 같이 획을 대표할 수 있는 특징들을 찾을 수 있다. 이와 같은 값들로 획의 특성을 파악하면 유사한 획들과 구별에 쓰일 수 있다.

그림 1은 쉽게 혼동 될 소지가 있는 북방 임(壬)자와 임금 왕(王)자의 획의 순서와 각 획마다 시작점을 나타내고 있다. '壬'자와 '王'자는 총 획수, 획의 순서가 같고 모양이 비슷해서 혼동될 수 있다. 하지만 '壬'자의 경우는 첫 획이 오른쪽에서 왼쪽 방향이고, '王'자는 오른쪽에서 왼쪽 방향이다. 이런 사실로 윈도우 좌표 시스템으로 표현하면 첫 획의 시작점 x좌표 값이 세 번째 획의 시작점 x좌표 값보다 큰지 작은지 알 수 있다. 세 번째 획의 시작점 x좌표 값에 상대적인 첫 획의 시작점 x좌표 값이 크면 '壬'자이고 작으면 '王'자로 구별할 수 있다.



북방 임(壬) 과 임금 왕(王)

① 첫 획의 시작점 ② 두 번째 획의 시작점

③ 세 번째 획의 시작점 ④ 네 번째 획의 시작점

그림 1. '壬'과 '王'의 획순과 시작점

2.2 방향코드를 이용한 구별

필기자로부터 입력된 데이터는 전처리과정을 거치고 특징 추출과정을 통해 획단위 특징 점들을 하나의 입력 벡터로 나타낸다. 각각의 입력벡터는 16방향 코드로 구성되며, 실제 획이 아닌 곳은 가상 획으로 16방향 코드를 부여한다[4].

16방향코드는 하나의 점을 중심으로 360도로 향할 수 있는 방향을 나타낸다. 온라인 필기 한자의 데이터는 2차원 좌표로 표현되어 16방향코드를 이용할 수 있다. 16방향코드는 두 개의 점에 대해서만 알면 구할 수 있기 때문에 한 획의 시작점과 끝점을 이용한 획 자체에 대한 방향뿐만 아니라 획 내의 점들 사의 방향변화와 획과 획 사이의 방향변화에도 이용할 수가 있다.

그림 1과 같은 혼동한자의 경우는 첫 획의 시작점 x좌표와 같이 2차원으로 표현되는 좌표 값들로 구별할 수 있지만 16방향코드를 이용하여 판단할 수도 있다. '壬'자는 첫 획이 오른쪽에서 왼쪽으로 향하고 '王'자는 왼쪽에서 오른쪽으로 향하고 있다. 획의 시작점과 끝점을 이용하여 각각을 16방향코드로 나타내면 그림 2와 같이 각각 0~1과 8~9의 방향코드로 표현 할 수 있다.

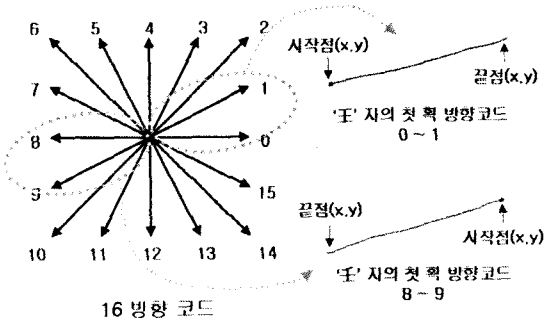


그림 2. '壬'과 '王'의 첫 획의 방향코드

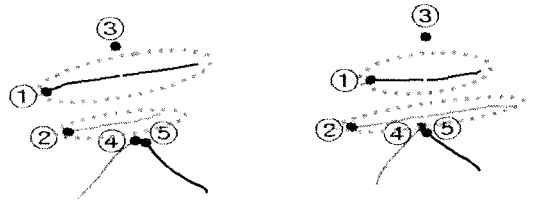
16방향 코드는 필기 한자의 획에 따라 약간의 차이가 날수 있다. '王'자의 첫 획의 경우 반듯하게 수평으로 쓰면 8이라는 방향코드를 얻을 수 있겠지만 필기자 개개인의 특성에 따라 시작점보다 끝점이 약간 올라간 형태로 방향코드 9를 얻을 수도 있다. 이 때문에 16방향코드를 조건화시켜 적용하려면 어느 정도의 가능한 상황까지 고려하여 범위를 정해 놓아야 한다.

16방향코드를 확장시켜 한 획의 끝점과 다음 획의 시작점 사이의 가상의 16방향 코드를 만들 수 있다. 가상의 16방향 코드는 실제 획의 끝점과 다음에 오는 획의 시작점을 가상적으로 이었을 때 나올 수 있는 16개의 방향코드를 말한다. 가상의 방향코드는 실제 16방향코드와 구분하기위해 16 ~ 31 방향코드를 할당한다.

2.3 획의 길이를 이용한 구별

한자의 획순과 방향코드가 같은 경우 이웃한 다른 획을 비교대상 기준으로 삼고 획의 길이를 보고 판단할 수 있는 구별 방법을 생각할 수 있다. 한자의 모양이 같은 끝 말(末)과 아닐 미(未)는 각각 획의 방향코드가 같고 획순이 일치하여 인식기가 혼동할 수 있는 한자이다.

획에서 시작점과 끝점은 알 수 있기 때문에 한 획의 넓이와 높이를 구할 수 있다. 넓이나 높이를 그 획의 전체 길이로 보고 판단 한다. 그림 3은 끝 말(末)과 아닐 미(未)의 두 번째 획에 대해 상대적인 첫 획의 길이로 구별하는 방법을 보여준다. 각각의 두 번째 획의 길이를 기준으로 삼아 첫 획의 길이가 상대적으로 크면 '末'자에 가깝고 짧으면 '未'에 가깝다고 본다.



Length ① > Length ②　　Length ① < Length ②

끝 말(末)　　아닐 미(未)

그림 3. '末'과 '未'의 첫 획의 길이

2.4 기타 구별 방법

앞서 소개한 점의 좌표, 16방향 코드, 획의 길이 구별은 기본적으로 혼동되는 한자들을 구별하는 결과검증 방법에 적용되고, 이 세 개의 방법으로 구별할 수 없는 한자들은 그 한자만이 가질 수 있는 고유한 특성을 조건화시켜 결과검증 방법에 적용 한다. 예를 들어 칼 도(刀)자와 힘 력(力)자의 경우 획순이 같다고 했을 때, 두 번째 획의 시작점 y값을 보고 판단 할 수 있다. 하지만 온라인 필기 한자에 있어 입력된 데이터가 확실하게 판단 할 정도의 y값을 기대할 수 없는 경우가 생길 수 있다. 이 때문에 임계값(Threshold)을 두어 첫 번째 획의 y최소값(ymin)에 비해 두 번째 획의 시작점 y좌표 값이 작다는 것을 판단하여야 한다. 임계값은 실험을 통하여 가장 적적할 값을 찾아 정한다. 이 밖에도 직선이 아닌 곡선의 획에서 특징 점을 찾아 구별하는 방법을 생각해볼 수 있다.

3. 결과검증 방법 구현 / 실험

3.1 결과검증 후처리

온라인 필기 한자 인식기는 필기자로부터 입력된 데이터와 인식기가 가지고 있는 모델사이의 거리(distance)를 계산한다. distance가 작을수록 모델과 유사성이 크다고 판단되어 최종 인식결과 한자가 될 수 있다. 각각의 후보한자들은 인식기에 의해서 distance값을 갖고 가장 작은 값을 1순위로 하여 10순위이하의 순위가 매겨진다. 이 10순위 후보한자들 중 혼동되는 한자가 있으면, 후처리로써 결과검증 방법을 거쳐 나온 점수가 혼동한자의 distance에 반영된다.

온라인 필기 한자를 인식과정에 있어 오인식은 문서의 왜곡(Distortion), 필기자의 다양한 필기형태 및 문서의 기울어짐, 인쇄상태의 불량 등 여러 가지 원인으로 정도의 차이가 있을 수 있으나 발생하기 마련이므로 인식 알고리즘을 개선하여 인식률을 높이는 것은 한계가 있다. 이러한 어려움을 해결하는 한 가지 바람직한 방법은 인식결과로 만들어진 텍스트를 실시간에 효율적으로 교정할 수 있는 후처리 알고리즘을 사용하는 것이다[5].

혼동한자 구별을 위한 결과검증 방법은 한자의 구조적 정보를 조건화시키기 위해 획을 구성하는 좌표들의 위치와, 획의 방향코드, 길이 등을 고려한다. 전처리 과정으로 입력된 한자는 정규화를 거치고 한자의 획에 대해 특징 점들을 추출하고 한자의 획 하나마다 방향코드를 계산한다.

그림 4는 '未'자에 대한 결과검증을 조건화시킨 함수(rv0514())를 보여준다. rv0514()에서는 '未'자의 조건으로 한자를 구성하고 있는 획들의 16방향코드와 '未'자와 구별하기 위해서 획의 길이를 본다. 필기자로부터 정확한 한자의 필순을 기대할 수 없는 경우가 생기기 때문에 가능한 다양한 필순에 대해서도 조건화 시킨다. 그림 4에서는 '未'자의 2획과 3획의 순서가 바뀔 경우를 고려하여 조건화 시켰다. 모든 조건의 검사가 끝나면 최종 인식결과에 반영되기 위해 가중치에 따라서 조건에 따른 차등된 점수(보너스/패널티)가 나온다. 보너스/패널티 점수는 ±5, ±10, ±20등으로 차등 적용하고 이 점수는 후보한자들 각각의 distance에 반영된다.

```
// '未' 자에 대한 결과검증 함수
function rv0514(...)
{
  ////////////////
     중 략
  ////////////////
  // '未' 와 구별하기 위해서 1획과 2획의 길이를 본다.
  first_stroke_length = first_stroke_xmax - first_stroke_xmin
  second_stroke_length = second_stroke_xmax - second_stroke_xmin
  // 획순서가 틀린 경우에 대비해서
  third_stroke_length = third_stroke_xmax - third_stroke_xmin
  // 올바른 획순
  if (first_stroke_dir_code is 15 or 0 or 1) and
    (second_stroke_dir_code is 15 or 0 or 1) and
    ////////////////
       중 략
    ////////////////
    (fifth_stroke_dir_code is 13 or 14 or 15) and
    (fifth_stroke_start_point_y+15 >= second_stroke_ymin) and
    ////////////////
       중 략
    ////////////////
    // 획의 길이 비교
    (second_stroke_length >= first_stroke_length+5))
    {
       return bonus2_score;
    }
  // 다른 획순(2획과 3획의 순서가 바뀐 경우)
  else if (first_stroke_dir_code is 15 or 0 or 1) and
          (second_stroke_dir_code is 15 or 0 or 1) and
          ////////////////
             중 략
          ////////////////
          (fifth_stroke_dir_code is 13 or 14 or 15) and
          (fifth_stroke_start_point_y+15 >= second_stroke_ymin) and
          ////////////////
             중 략
          ////////////////
          // 획의 길이 비교
          (third_stroke_length >= first_stroke_length + 5))
          {
             return bonus2_score;
          }
  // 조건에 만족하지 못하면 약간의 페널티 적용
  else
  {
     return penalty1_score;
  }
}
```

그림 4. '未'의 결과검증 함수

3.2 실험 결과

온라인 필기 한자 인식기 후처리부분에 결과검증 방법의 적용여부에 따라 나온 에러율을 구하였다. 인식기가 최종 인식결과를 산출하기 전에 나온 10순위 후보 한자에 대해 결과검증 방법을 적용시켰다. 결과검증을 위한 한자는 2획에서 10획 사이의 혼동되기 쉬운 226개의 한자로 구성하였고 각각의 한자에 대해 구조적 정보를 조건화 시켜 함수로 만들었다.

인식기가 최종 인식결과를 산출하기 전 10순위까지의 후보 한자에 대해서 혼동되는 226개의 한자 중 포함된 한자가 있으면 후처리로써 결과검증 함수를 실행시켜 보너스 또는 패널티 점수를 매긴다. 보너스/패널티 점수는 한자 고유의 조건에 얼마나 부합하는지에 따라 차등 적용하였다. 보너스/패널티 점수를 너무 크게 주면 인식기의 인식결과가 무시되므로 작은 값을 차등 적용시킨다.

그림 5는 결과검증 방법을 통해 1순위 후보가 바뀜으로써 최종 인식 한자가 바뀐 예를 보여준다. 결과검증 방

법을 사용하지 않았을 경우 최종인식 한자는 끝 말(末)자가 되었다. '末'자는 첫 획의 길이가 두 번째 획의 길이(넓이)에 비해 크고 첫 획의 시작점 x값이 두 번째 획의 시작점의 x값보다 작다. '末'자는 '未'자와 반대의 경우이다. 필기자로부터 입력된 데이터는 '未'자에 가깝기 때문에 필기자가 의도한 한자는 아닐 미(未)자로 판단할 수 있다. '末'자로 오인식되는 경우를 막기 위해 결과검증 방법을 인식 후처리에 적용하면 '末'자는 패널티 점수(+5점)를 가지게 되고 '未'자는 보너스 점수(-10점)를 얻게 되어 각각의 후보한자 distance값에 더해진다. '末'자는 10이 감소되어 distance값이 14 → 4로 줄어들었고 '未'자는 5점이 더해져서 10 → 15로 늘어났다. 결국 1순위 후보는 distance값이 작은 '未'자로 바뀌게 된다.

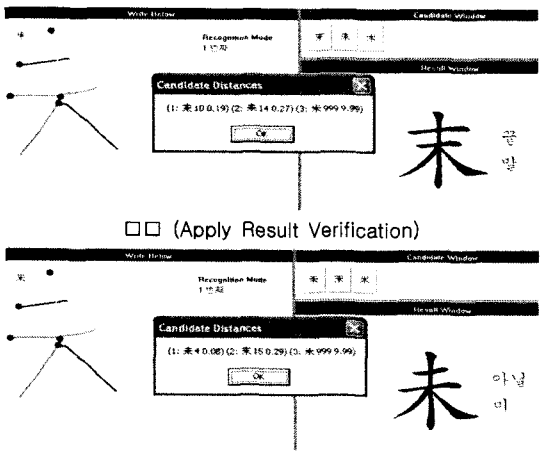


그림 5. 인식 결과검증 적용

모델과 테스트 데이터 모두 한 사람당 약 2,300여자의 한자를 수집하였다. 인식기의 모델은 남자 11명과 여자 9명으로 총 20명의 46,560개의 한자를 훈련하여 만들었고 테스트 데이터는 남자 4명과 여자 6명으로 총 10명의 22,856개의 한자로 구성하였다. 테스트 데이터는 22,856개의 전체한자(All Data)와 전체한자 중 1,970개의 혼동한자(Confusion Data)를 따로 분리하였다.

표 1은 인식기 후처리부분에 결과검증 방법의 적용 유무에 따른 각각의 테스트 데이터의 정인식률, 거부율, 오류율을 나타낸다.

표 1. 결과검증(Result Verification) 실험

(한자수)

RV / Test Data	RV Off			RV On		
	정인식률	거부율	에러율	정인식률	거부율	에러율
All (22,856)	96.03% (21,948)	2.48% (566)	1.50% (342)	96.22% (21,991)	2.42% (554)	1.36% (311)
Confusion (1,970)	98.68% (1,944)	0.56% (11)	0.76% (15)	99.39% (1,958)	0.20% (4)	0.41% (8)

전체 테스트 데이터와 혼동데이터 모두 거부율과 에러율이 줄어들어 인식률을 높아졌다. 테스트 데이터에 혼동되는 한자가 얼마나 많이 포함되는지에 따라 에러율과 거부율은 더 감소될 수 있다.

결과검증 방법의 경우는 어디까지나 온라인 필기 한자 인식기의 최종 인식결과 산출에 보조역할로 도움을 준다. 만약 결과검증 방법이 인식기가 인식한 한자를 무시할 정도로 크게 영향을 미치면 오히려 인식결과가 나빠지기 때문에 인식기의 후처리로서만 작용해야 한다.

실험에서는 인식기가 최종적으로 인식결과를 산출하기 전 후보한자에 대한 결과검증 방법을 적용하였다. 이는 최종 인식 한자에 대해 다시 한 번 확인하는 과정이기 때문에 인식기의 최종 인식결과를 더 신뢰할 수 있다.

4. 결 론

본 논문에서는 온라인 필기 한자인식의 후처리에 결과검증 방법을 적용시킴으로써 쉽게 혼동될 수 있는 한자들을 구별하였다. 결과검증 방법은 2획부터 10획 이하의 혼동 한자들에 대해서 각각의 한자의 구조적 정보를 조건화시켜 함수로 만들고 입력된 데이터를 검사한다.

결과검증 방법은 휴리스틱을 이용하여 사람의 눈으로 쉽게 구별할 수 있는 한자의 특징을 찾아내어 조건화 시켰다. 이 결과 인식기가 혼동할 수 있는 한자들을 구별하는데 도움을 줄 수 있었다. 11획 이상으로 이루어진 한자에 대해서도 한자의 구조적 정보를 조건화 시키면 혼동한자 구별을 위한 결과검증 방법을 적용할 수 있다.

향후연구 과제로 혼동 한자들에 쓰이는 결과검증 방법을 휴리스틱이 아닌 다른 분류방법의 사용을 생각해볼 수 있다. 휴리스틱방법의 경우 한자 하나에 대해서 고유한 특성을 조건화시켜야 하는 불편함이 있다. 구별해야 할 한자들이 많아질수록 각각의 한자에 대해서 휴리스틱으로 고유한 특성을 파악하여 조건화 시켜야 하기 때문에 비효율적일 수 있다. 이와 같은 점을 개선하기 위해서 SVMs(Support Vector Machines)와 같은 분류 방법을 이용할 수 있다. 혼동한자는 대부분 한 개 이상의 유사한 한자들과 관련지을 수 있어 휴리스틱이 아닌 SVMs와 같이 이진분류에 좋은 성능을 보이는 방법을 고려할 수 있다.

5. 참고 문헌

[1] 김상균, 이종국, 김항준, "은닉 마르코프 모델과 레벨 빌딩 알고리즘을 이용한 흘림체 한자의 온라인 인식", *경북대 공대 전자기술연구지*, 제 17권 2호 pp. 62-69, 1996.

[2] 전상엽, 권희용, "DP 정합을 이용한 필기체 한자 인식", *한국멀티미디어학회 04 춘계학술발표대회논문집*, pp. 285-288, 2004.

[3] 진원, 김기두 "유닛 재구성 방법을 이용한 PDA용 온라인 필기체 한자인식", *전자공학회논문지*, 제 39권, SP편, 1호, pp. 97-107, 2002.

[4] 박재성, 송영길, 이향미, 박진열, 이은주, 김태균, 획 방향과 길이요소 가중 DP 매칭에 의한 흘려 쓴 한글 인식", *인공지능신경망 및 퍼지시스템 종합학술대회 논문집*, pp. 170-179, 1992.

[5] 조완현, "문자인식의 소개 및 새로운 인식시스템의 개발", *Proceedings of the Spring Conference*, Korean Statistical Society, 1998.