# 한글 문서인식의 오인식 수정에 관한 연구

*박 진 규, 김 진 형

한국과학기술원, 전산학과

# A Study on the Error Correction in HANGUL Text Recognition

Jinkyu Park and Jin H. Kim

KAIST, Department of Computer Science

## 요 약

정보처리 자동화를 위한 문서 인식 시스템의 오인식 수정을 위한 후처리 알고리즘은 사용되는 문맥적 지식의 표현 방법에 따라 하향식, 상향식 그리고 복합적 방법으로 나뉘어 질 수 있다.

본 논문에서 제안할 오인식 수정 알고리즘은 하향식 방법인 Modified Viterbi 알고리즘과 상향식 방법인 Dictionary Look-Up 알고리즘을 한글 문장의 띄어쓰기 단위인 어절의 특성에 맞게 혼합한 복합적 방법이다

문자간의 혼동확률과 어절을 이루는 각 형태소의 사전확률(prior probability) 및 형태소간의 결합 가능성을 나타내는 grammar를 이용하여 어절의 끝에서부터 파싱(parsing)하면서 가장 확률이 높은 입력 어절을 찾는다

## I. 서 론

현대 사회의 정보화 및 컴퓨터 산업의 급속한 발달로 인하여 정보처리 자동화에 대한 요구가 급증하고 있다. 60 년대 이후, 정보처리 자동화를 위하여 광학 문자 인식기(OCR)에 의한 자동 문서 인식에 관한 많은 연구가 진행되어 현재는 몇 가지의 제한된 인쇄체 영어 문서나 제한된 형태의 필기체 문서를 인식하는 시스템이 실용화 단계에 있다.

문서 인식은 형태 인식(pattern recognition)의 한 분야로 scanner를 통하여 입력된 문서를 받아 잡음(noise) 제거, 문자 영상의 추출, 표준화, 세선화 등을 수행하는 전처리(preprocessing) 과정과 문자를 인식하는 문자 인식 과정, 문맥적 지식(contextual knowledge) 및 문자 인식 방법의 특성을 이용한 오인식 수정 또는 후처리(postprocessing) 과정으로 이루어 진다[1]. 자동 문서 인식 시스템의 인식 과정은 입력 문자들의 유사성 및 입력 문자에 포함된 잡음으로 인하여 입력 문자를 항상 정확하게 인식하지는 못한다 따라서 오인식된 문자를 후처리 과정에서 수정할 수 있어야 신뢰성 있는 문서 인식 시스템을 구현할 수 있다 사람의 문서 인식에서 사용되는 문맥적 지식은 단어내 문자의 연결 관계, 단어간의 연결관계, 문장의 구조, 문서의 주제 등을 들 수 있으나 컴퓨터로 구현하기에는 많은 어려움이 뒤따라 기존의 많은 후처리 알고리즘은 단어내 문자의 연결 관계만을 문맥적 지식으로 사용하여 단어별로 오인식 수정을 하고 있다 문자 인식 방법의 특성으론 어떤 문자를 그 문자 혹은 다른 어떤 문자로 인식할 수 있는 혼동확률(confusion probability)을 들 수 있는데 이는 각 방법의 특성에 따라 달라질 수 있다

오인식 수정을 위한 후처리 알고리즘은 문맥적 지식의 표현 방법에 따라 크게 세가지로 나눌 수 있다. 첫째는 문맥적 지식의 구조적 표현에 기초한 하향식(top-down) 방법이고 둘째는 문맥적 지식의 통계적(확률적) 표현에 기초한 상향식(bottom-up), 셋째는 상향식과 하향식을 결합한 복합적(hybrid) 방법이다 문맥적 지식의 구조적 표현이란 사전(dictionary), 구문(syntax), 어의(semantics) 등을 개략적 혹은 명시적으로 나타내는 것이다 개략적으로 나타낸 방법의 예로는 적법한 문자 조합을 이진 배열로 구성한 Binary n-gram[2] 알고리즘이 있고 명시적으로 나타낸 방법의 예로는 사용되는 모든 단어를 사전으로 구성한 Dictionary Look-Up(DLA)[3, 4] 알고리즘이 있다 문맥적 지식의 통계적 표현이란 단어 형성의 m차 Markov 가정에 의한 전이확률(transitional probability)로 이 방법의 대표적인 예로는 Modified Viterbi 알고리즘(MVA)[5]이 있다 복합적 방법으론 DLA와 MVA를 혼합한 Predictor-Corrector 알고리즘(PCA)[6]이 있다 DLA나 PCA는 수정률은 좋으나 사용되는 모든 단어의 사전 구성으로 인하여 많은 비용이 소요되고, MVA는 비용은 적으나 수정률이 떨어진다[4]

본 논문에서 제안할 오인식 수정을 위한 후처리 알고리즘은 하향식 방법인 DLA와 상향식 방법인 MVA를 한글 문장의 띄어쓰기 단위인 어절의 특성에 맞게 혼합한 복합적 방법이라 하겠다 본 알고리즘은 수정된 Viterbi 알고리즘의 문자간 전이확률대신 수정 단위인 어절을 형태소로 분류하여 각각의 사전확률(prior probability)과 함께 사전에 저장하고, 한 어절을 이룰 수 있는 형태소간의 결합 관계는 grammar 형태로 나타낸다 한 어절을 인식하였을 때 문자간의 혼동확률과 각 형태소의 사전확률 및 결합 가능성을 나타내는 grammar를 이용하여 어절의 끝에서부터 파싱(parsing)하면서 가장 확률이 높은 입력 어절이 무엇인가를 찾는다.

본 논문의 구성은 다음과 같다 II장에서 기존의 알고리즘 중 확률론을 이론적 배경으로 하는 DLA, MVA, PCA 각각에 대하여 알아보고, III장에서 한글 문장의 특성[7]을 어절 중심으로 기술한다 IV장은 제안된 알고리즘을 설명하고, V장의 결론으로 논문을 맺는다

## II. 기존의 오인식 수정 알고리즘

서론에서 언급한 오인식 수정을 위한 알고리즘 중 하향식(top-down) 방법의 하나인 Binary n-gram 알고리즘을 제외한 대부분의 오인식 수정 알고리즘은 확률론을 이론적 배경으로 삼고 있으며, IV에서 제시하고자 하는 알고리즘 역시 확률론에 기초한다 따라서 본 장에서는 확률론을 이론적 배경으로 하는 하향식의 DLA, 상향식의 MVA, 복합적의 PCA 각각에 대하여 알아본다

### 2.1 이론적 배경

확률론에 기초하는 알고리즘은 문자 인식 알고리즘이 어떤 글자를 받아들여 그 글자 혹은 다른 어떤 글자라고 확정적으로 인식하는 경우, 인식 단어를 야기시킨 가장 확률이 높은 입력 단어가 무엇인가를 찾는다

인식 단어를 $X = X_0X_1 \quad X_nX_{n+1}$이라 하자 $X_0$와 $X_{n+1}$은 n개의 글자로 이루어진 단어를 앞뒤 단어와 구분하여 주는 구분 문자로 여백, 특수 부호 등을 나타낸다 인식 단어가 $X = X_0X_1 \ . \ X_nX_{n+1}$이었을 때 입력 단어가 $Z = Z_0Z_1 \quad Z_nZ_{n+1}$이었을 확률은 Bayes의 이론에 의하여

$$P(Z|X) = P(X|Z) * P(Z) / P(X) \qquad (1)$$

이 된다 P(Z)은 Z의 사전확률(prior probability)을, P(X)는 스트링 X의 확률을 나타낸다 P(X)는 Z와는 독립적이므로 P(Z|X)를 가장 크게 하는 입력 단어 Z는

$$G(X,Z) = P(X|Z) * P(Z) \qquad . \quad (2)$$

를 가장 크게 하는 Z가 된다 컴퓨터의 메모리에 가능한 모든 P(X|Z)를 저장하는 것은 X와 Z의 너무나 많은 조합으로 인하여 불가능한 일이다 따라서 $X_0$, $X_1$, , $X_n$, $X_{n+1}$ 간에 조건부 독립(conditional independence)을 가정하면 P(X|Z)는

$$\begin{aligned} P(X|Z) &= P(X_0X_1 \ .. \ X_nX_{n+1}|Z_0Z_1 \quad Z_nZ_{n+1}) \\ &= P(X_0|Z_0 \ . \ Z_{n+1}) \ .. \ P(X_{n+1}|X_0 \quad X_nZ_0 \quad Z_{n+1}) \\ &= P(X_0|Z_0) \ P(X_1|Z_1) \ . \ P(X_{n+1}|Z_{n+1}) \\ &= P(X_1|Z_1) \ . \ P(X_n|Z_n) \end{aligned}$$

이 된다 $X_0$, $X_1$, . , $X_n$, $X_{n+1}$ 간의 조건부 독립 가정은 인쇄체 문서나 가능하고 필기체 문서에서는 한 문자의 형태가 다음에 나오는 문자의 형태에 영향을 줄 수 있으므로 이러한 가정은 성립하지 않는다 여기서 $P(X_i|Z_i)$는 문자 $Z_i$를 문자 $X_i$로 인식할 혼동확률(confusion probability) 을 나타내는 것으로 문자 인식 알고리즘의 특성에 따라 달라지게 된다

따라서 구하고자 하는 Z는 (2)식에 logarithm을 취한

$$G(X,Z) = \sum_{i=1}^{n} \log P(X_i|Z_i) + \log P(Z) \qquad .. \quad (3)$$

에 의하여 구할 수 있다 P(Z)은 문맥적 지식을 나타내는 것으로 P(Z)을 표현하는 방법에 따라 오인식 수정 알고리즘은 구분되어 진다

### 2.2 Dictionary Look-Up 알고리즘(DLA)

DLA는 가장 먼저 나온 오인식 수정 알고리즘으로 여기서는 Bledsoe, Browning 알고리즘[3]의 확장인 Toussant[4]의 방법을 기술한다

사용되는 모든 단어를 사전으로 구성하여, 길이가 n인 인식 단어 X를 야기시킨 입력 단어 Z는 사전에 저장된 길이 n의 모든 단어에 대하여 2 1의 (3) 식을 다시 쓴 식 (4)를 계산하여 그 중 최대값을 갖는 것으로 한다

$$S(Z) = \sum_{i=1}^{n} \log P(X_i|Z_i) + \log P(Z) \qquad (4)$$

DLA는 수정률은 좋으나 사용되는 모든 단어를 사전에 저장하여야 하고 또한 같은 길이의 모든 단어에 대하여 (4) 식을 계산하이야 하므로 많은 메모리와 시간이 필요하다[6]

### 2 3. Modified Viterbi 알고리즘(MVA)

사용되는 모든 단어에 대하여 P(Z)를 사전에 저장하는 DLA의 단점을 극복하기 위하여 MVA에서는 (3) 식에서의 P(Z)를 단어 형성의 m차 Markov 가정을 사용하여

$$P(Z) = P(Z_{n+1}|Z_{n+1\,m} \quad Z_n) \quad P(Z_1|Z_0)$$

으로 근사시킨다 $P(Z_i|Z_{i-m} \quad Z_{1\,1})$은 전 m개의 문자가 $Z_{i\,m}$ $Z_{1\,1}$이었을 때 문자 $Z_i$가 나올 확률로 m차 전이확률이라 한다 만약 m = 1이면 $P(Z) = P(Z_{n+1}|Z_n) \quad P(Z_1|Z_0)$이 되어 (3) 식은

$$G_1(X,Z) = \sum_{i=1}^{n} \log P(X_i|Z_i) + \log P(Z_i|Z_{1\,1})$$

이 된다

MVA는 문맥적 지식을 m차 전이확률로 나타내므로 시간과 메모리 면에서 DLA보다 적게 드나 수정률은 떨어진다[6]

### 2.4. Predictor-Corrector 알고리즘(PCA)

PCA는 하향식의 DLA와 상향식의 MVA를 결합한 복합적 방법으로 Predictor 단계와 Corrector 단계로 이루어 진다

Predictor 단계

구문구조 $Z_{n+1}$ 1 MVA를 사용하여 단어 Z를 찾는다

2 단어 Z의 VALUE를 구한다

$$VALUE(Z) = \sum_{i=1}^{n} \log P(Z_i|Z_{i-1})$$

3 이진 검색으로 VALUE(Z)가 사전에 있는가를 본다 만약 있으면 Z를 출력하고 없으면 Corrector 단계로

Corrector 단계

4 같은 길이의 모든 단어에 대하여 S(Z)을 구한다

$$S(Z) = \sum_{i=1}^{n} \log P(X_i|Z_i) + \log P(Z)$$

5 S(Z)을 최대로 하는 Z를 출력한다

위의 3에서 알 수 있는 바와 같이 PCA는 길이가 같은 어떤 두 단어도 같은 VALUE를 갖지 않는다는 가정을 사용하고 있다 이 알고리즘은 시간이 적게 드는 Predictor 단계에서 실패를 하였을 경우에만 많은 시간을 요하는 Corrector 단계를 수행하여 평균 수행시간을 줄인 것으로 인식률은 DLA와 같은 수준이다.

## III. 한글 문장의 구조적 특성

한글은 계통적으로는 우랄 알타이어에, 형태 및 어법상으로는 교착어(혹은 첨가어)에 속한다 따라서 다른 언어와는 달리 어순이 자유스럽고 생략이 많으며 용언의 활용과 어미, 조사의 다양한 변화로 어절의 형태가 매우 복잡하다. 어절이란 문장 성분의 최소 단위로서 띄어쓰기 원칙을 따른다 하나의 어절은 하나 이상의 형태소의 결합으로 이루어지는데, 형태소는 의미, 기능에 따라 실질 형태소와 형식 형태소로 나눌 수가 있다 실질 형태소는 한 어절안의 중심이 되는 의미를 표시하는 형태소로 체언, 수식언, 감탄사, 용언의 어근 등이 이에 속하며 형식 형태소는 실질 형태소에 결합되어 말과 말 사이의 관계를 형식적으로 표시하는 형태소로 조사, 어말어미, 접사, 시제어미 등이 이에 속한다

어절을 중심으로 한 한글의 특성은 다음과 같다

(1) 실질 형태소 다음에 여러개의 형식 형태소가 올 수 있으며, 또한 생략 될 수도 있다.
예) 교사-(이)-지-만

(2) 합성어가 많고, 띄어쓰기가 일정하지 않다.
예) 민족중흥 -- 민족 중흥

(3) 용언의 활용과 어미, 조사의 변화가 다양하다
예)학생 + 은(도, 만, 과, 에게, )

(4) 모음충돌 회피, 모음 탈락, 음절 탈락 등의 음운 현상이 일어난다
예) 사이 ---> 새

위에서 살펴본 한글 문장의 특성상 한글은 어절을 이루는 형태소 결합의 문법적 규칙을 정확하게 설정하기는 매우 어렵다

## IV. 제안된 오인식 수정 알고리즘

MVA와 하향식 방법인 DLA를 한글 문장의 구조적 특성에 맞게 혼합한 복합적 방법이라 하겠다 본 알고리즘은 MVA의 문자간 전이확률대신 어절을 형태소로 분류하여 각각의 사전확률(priori probability)과 함께 사전에 저장하고, 한 어절을 이룰 수 있는 형태소간의 결합 관계는 grammar 형태로 나타낸다 한 어절을 인식하였을 때 문자간의 혼동확률과 각 형태소의 사전확률 및 결합 가능성을 나타내는 grammar를 이용하여 어절의 끝에서부터 파싱(parsing)하면서 가장 확률이 높은 입력 어절이 무엇인가를 찾아낸다.

4 1에서는 기존의 알고리즘을 한글에 적용할 경우 발생할 문제점을 살피고 4 2는 어절을 이루는 형태소간의 결합관계 및 사전 구성 방법에 관하여 기술하고 이에 따른 알고리즘은 4 3에 주어진다

### 4.1 기존 방법의 한글 적용시 문제점

앞에서 살펴본 한글 문장의 특성을 고려하여 기존의 방법들을 한글 문장에 적용할 경우 발생하는 문제들을 간략히 기술한다.

DLA나 PCA는 사용되는 모든 단어를 사전에 저장한다 한글은 조사, 어미의 발달이 현저하고 복합어가 많아 어절 단위로 사전을 구성하여 DLA나 PCA를 적용하는 것은 사전의 크기가 방대하여지므로 불가능한 일이다.

MVA는 P(Z)를 단어 형성의 m차 Markov 가정에 의하여

$$P(Z) = \{P(Z_{n+1}|Z_{n+1-m} \quad Z_n) \; .. \; P(Z_1|Z_0) \; P(Z_0)\}$$

로 근사시킨다 굴절어(영어, 불어, 독어 등)나 고립어(중국어 등)와 같이 띄어쓰기 단위가 독립된 단어 하나이고, 띄어쓰기가 정확히 지켜지는 언어에서는 P(Z)를 MVA에서와 같이 단어 형성의 m차 Markov 가정에 의하여 근사시킬 수 있다 하지만 한글에 m차 Markov 가정을 사용하여 전이확률을 구한다면 띄어쓰기가 일정하지 않아 신뢰성 있는 전이확률을 구할 수 없고, 문자 수가 많아 m > 1인 경우 방대한 메모리가 필요하게 된다

### 4.2. 사전 구성

본 장에서는 III과 4-1을 고려하여 한글 문서의 오인식 수정에 효율적인 문맥적 지식(contextual knowledge)을 제공하기 위한 사전 구성에 관하여 기술한다 한글은 어절을 이루는 형태소 결합의 문법적 규칙을 정확하게 설정하여 컴퓨터에 표현하기는 매우 어려우나 어절의 형태를 다음과 같이 나타낼 수는 있다.

< 어절 형태 >

1 관형사, 부사, 체언, 감탄사 등이 독립적으로 사용
2 체언, 부사 + 조사
3 체언 + 어미
4 체언 + 조용사 + (시제어미) + 어미
5 체언 + 조용사 + (시제어미) + 명사형어미 + (조사)
6 용언 + (어미 + 보조용언) + (시제어미) + 어미
7 용언 + (어미 + 보조용언) + (시제어미) + 명사형어미 + (조사)

조용사라는 것은 학자들에 따라 여러가지 의미로 쓰이지만 여기서는 명사에 붙어 용언을 이루는 것들을 총칭한다 예를 들어 "이다", "하다", "스럽다", "답다" 등이 여기에 속한다 괄호는 생략 가능함을 나타내며 체언, 시제어미, 조사는 겹쳐서 나올 수 있다 결합 가능한 {어미 + 조사}도 하나의 어미로 나타내었다

위에서 구분한 어절 형태를 어절의 끝에서부터 파싱(parsing)할 수 있도록 grammar로 나타내면 <표 1>과 같다

S ---> 어미+A | 조사+B | 명사형어미+C | 관형사
| 부사 | 감탄사
A ---> 보조용언+D | 시제어미+E | 조용사+F | 용언 | 체언
D ---> 어미+G
G ---> 용언
E ---> 보조용언+D | 조용사+F | 용언
F ---> 체언+H
H ---> 체언+H | ^
B ---> 체언+H | 조사+B | 명사형어미+C | 부사
C ---> 시제어미+E | 보조용언+D | 조용사+F | 용언

<표 1>

위의 grammar에서 영문자 및 각 형태소는 *non-terminal* 심볼을 나타내며, 각 형태소에 속하는 *terminal* 심볼은 생략되었다. *start* 심볼은 S이다. 어절을 끝에서부터 파싱할 수 있도록 grammar를 구성한 것은 조사나 어미 처리를 쉽게 하기 위함이다

사전에 저장되는 단위는, grammar의 *terminal* 심볼을 원칙으로 하되 단일어가 결합된 복합어는 각 단일어로 분리하고 {시제어미 + 시제어미}, {조용사 + 어미}, {조사 + 조사}, {조용사 + 시제어미} 등의 형태소 결합에서 자주 쓰이는 것은 하나로 저장한다. 또한 변형된 어간이나 어미도 저장한다 이는 소요되는 메모리의 양과 사전 검색 시간을 줄일 수 있기 때문이다 앞으로는 사전에 저장된 한 단위를 토큰(token)이라 부른다

## 4.3. 오인식 수정 알고리즘

II의 (3) 식

$$G(X,Z) = \sum_{i=1}^{n} \log P(X_i|Z_i) + \log P(Z)$$

에서 모든 Z에 대하여 P(Z)를 저장하는 것은 메모리의 문제를 야기하므로 MVA에서는 문자간의 m차 Markov 가정에 의하여 근사시키지만 본 알고리즘에서는 어절의 구조적 특성에 맞게 다음과 같이 근사시킨다

n개의 문자로 이루어진 어절 Z가 4.2의 grammar를 만족하는 m, 1 <= m <= n, 개의 토큰 $T_1$, ., $T_m$으로 나누어지고 결합 가능한 토큰간의 조건부 독립이 성립한다고 하면 P(Z)은

$$P(Z) = P(T_1 \cdot T_m)$$
$$= P(T_1) P(T_m|T_m) \quad P(T_i|T_{i-1} \quad T_1)$$

조건부 독립에 의하여

$$= P(T_m) \quad P(T_1)$$

로 근사시킬 수 있다 따라서 구하고자 하는 Z는 4.2의 grammar를 만족하는 Z 중 다음 식을 가장 크게 하는 Z가 된다

$$G(X,Z) = \sum_{i=n}^{1} \log P(X_i|Z_i) + \sum_{i=m}^{1} \log P(T_i)$$

## V. 결 론

정보처리 자동화를 위한 자동 문서 인식 시스템은 정확한 문서 인식을 위하여 효율적인 문자 인식 알고리즘 뿐만 아니라 수정률이 뛰어난 오인식 수정 알고리즘이 요구된다 문서 인식의 오인식 수정을 위한 후처리 알고리즘은 DLA나 PCA에서와 같이 사용되는 문맥적 지식이 많을 수록 수정률은 향상되나 비용이 많이 들고, MVA에서와 같이 문맥적 지식이 적으면 비용은 적게 드나 수정률이 떨어진다. 한글은 다른 언어와는 달리 용언의 활용과 조사, 어미의 다양한 변화로 어절의 형태가 매우 복잡하여 수정률이 좋은 DLA나 PCA를 사용하기에는 소요되는 많은 메모리와 시간으로 인하여 불가능하게 된다 따라서 한글의 특성에 맞는 수정률이 뛰어난 오인식 수정 알고리즘이 요구된다 본 논문에서 제시한 알고리즘은 문자간의 혼동확률과 어절을 이루는 각 형태소의 사전확률 및 형태소간의 결합 가능성을 나타내는 grammar를 이용하여 오인식 수정을 하므로써 DLA나 PCA보다 적은 비용으로도 MVA보다는 높은 수정률을 얻을 수 있으리라 기대된다

## 참 고 문 헌

[1] J J Hull & S N Srihari, "Experiments in Text Recognition with Binary n-Gram and Viterbi Algorithm", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol PAMI-4, PP 520-530, 1982

[2] E M Riseman & A R Hanson, "A Contextual Postprocessing System for Error Correction Using Binary n-Gram", *IEEE Transaction on Computers*, Vol C-23, PP 480-493, 1974

[3] W W Bledsoe & J Browning, *Pattern Recognition*, PP 301-306, Wiley, New York, 1966

[4] G T Toussaint, "The Use of Context in Pattern Recognition", *Pattern Recognition*, Vol 10, PP 189-204, 1978

[5] R Shinghal & G T Toussaint, "Experiments in Recognition with the Modified Viterbi Algorithm", *IEEE Pattern Analysis and Machine Intelligence*, Vol PAMI-1, PP 184-193, 1979

[6] R Shinghal & G T Toussaint, "A Bottom-Up and Top-Down Approach to Using Context in Text Recognition", *Int J Man-Machine Studies*, Vol 11, PP 201-212, 1979

[7] 조 규빈, 하이라이트 고교 문법, 지학사, 1986