음절 Bi-gram정보를 이용한 한국어 OCR 후처리용 자동 띄어쓰기

전남열° 박혁로
전남대학교 전산학과
{nyjeon, hrpark@cs.chonnam.ac.kr}

Automatic Word-Spacing of Syllable Bi-gram Information for Korean OCR Postprocessing

Nam-Youl Jeon° Hyuk-Ro Park
Dept. of Computer Science, Chon-Nam National University

요 약

문자 인식기를 가지고 스캔된 원문 이미지를 인식한 결과로 형태소 분석과 어절 분석을 통해 대용량의 문서 정보를 데이터베이스에 구축하고 전문 검색(full text retrieval)이 가능하도록 한다. 그러나, 입력문자가 오인식된 경우나 띄어쓰기가 잘못된 데이터는 형태소 분석이나 어절 분석에 그대로 사용할 수가 없다. 한글 문자 인식의 경우 문자 단위의 인식률은 약 90.5% 정도나 문자 인식 오류와 띄어쓰기 오류 등을 고려한 어절 단위의 인식률은 현저하게 떨어진다. 이를 위해 한국어의 음절 특성을 고려해서 사전을 기반하지 않고 학습이 잘된 말뭉치(corpus)와 음절 단위의 bi-gram 정보를 이용한 자동 띄어쓰기를 하여 실험한 결과 학습 코퍼스의 크기와 띄어쓰기 오류 위치 정보에 따라 다르지만 약 86.2%의 띄어쓰기 정확도를 보였다. 이 결과를 가지고 형태소 분석과 언어 평가 등을 이용한 문자 인식 후처리 과정을 거치면 문자 인식 시스템의 인식률 향상에 크게 영향을 미칠 것이다.

1. 서론

대용량의 문서정보 데이터베이스를 서비스하기 위해서는 문서내용을 대표할 수 있는 키워드를 추출하여 저장함으로써 사용자가 키워드를 통하여 필요한 정보에 접근할 수 있도록 하여야 한다.

데이터베이스에 축적되는 여러 문서 정보 중에서 워드 프로세서 파일을 중간 포맷 형태인 DVI(Device Independent Format)이나 PDF(Portable Document Format)등으로 변환하여 키워드를 추출하는 방법과 워드프로세서 파일이 존재하지 않으면 원문정보를 이미지로 스캔한 후 서지 정보를 사람이 입력하여 검색에 사용하는 수동 입력 방식이 일반적으로 사용되었다. 이러한 수동 입력에 기반 한 정보서비스는 전문 검색(full text retrieval)이 불가능하여 문서의 크기가 클 경우 사용자가 문서내의 특정한 정보를 얻기가 매우 힘들며 초기 서지정보 입력 비용이 많이 소요되는 단점이 있다. 이러한 단점을 극복하기 위한 방법으로는 문자 인식 기술을 이용한 정보 서비스 방식을 들 수 있다[1].

이 서비스는 스캔된 원문 이미지를 문자 인식기의 입력으로 사용하여 문자 정보를 추출한 후, 이 문자 정보를 정보검색 경로로 사용한다. 이러한 방법을 이용하면 원문 이미지로부터 문자 정보를 추출하기 때문에 페이지 단위의 전문 검색이 가능하며, 자동화된 문자 인식 프로그램 을 이용하기 때문에 입력 비용 문제도 동시에 해결된다.

문자 인식기를 이용한 정보 서비스는 문자 인식기의 성공률에 매우 의존적이다. 영어권 언어는 99%이상이 인식성공률을 보여 이를 이용한 정보서비스에는 문제가 없으나 한국어의 경우 문자 단위의 인식률은 약 90.5%정도로 알려져 있다[2]. 그러나 이 성공률은 문자 단위로 계산을 한 것이기 때문에 띄어쓰기 등의 문자 인식 오류를 포함하고 있지 않다. 따라서 문자 인식 오류 및 띄어쓰기 오류를 모두 고려해야 하는 어절 단위의 인식 성공률은 문자 단위의 인식률보다 훨씬 낮다.

본 논문에서는 문자 인식기의 어절 단위의 인식률을 높이기 위해 문자 인식기의 띄어쓰기 오류를 회복할 수 있는 방법에 대해 연구한다. 이를 위하여 문자 인식기에 따른 오류 유형과 각각의 발생 빈도를 조사하고, 그 중에서

띄어쓰기 오류에 대한 문제를 해결하기 위해서 사전에 기반하지 않고 원형코퍼스(raw corpus)로부터 필요한 음절 bi-gram 정보를 이용한 한국어 OCR 후처리용 자동 띄어쓰기를 제시한다.

본 논문의 구성은 서론에 이어 2장에서는 문자 인식 시스템 후처리에 관련된 연구와 한국어 자동 띄어쓰기 관련 연구를 살펴보고, 3장에서는 문자인식 오류를 유형별로 분류하고 자동 띄어쓰기 시스템을 제안하고, 4장에서는 본 논문에서 제안한 방법을 KT 테스트 컬렉션[10]에 적용한 실험 결과를 기술한다. 마지막으로 5장에서 결론과 향후 연구 방향에 대하여 기술한다.

2. 관련연구

정보 검색 관점에서 문자 인식기가 생성한 오류를 처리하는 방법은 크게 두 가지로 나누어진다. 첫 번째 방법에서는 문자 인식기의 오류를 수정하지 않고 검색 모델을 유연하게 확장함으로써 비록 색인에 오류가 포함되어 있다고 하더라도 검색이 될 수 있도록 하는 방법이다. 또 다른 방법으로는 언어정보를 이용하여 문자 인식기의 오류를 수정하는 방법이 있다.

2.1 정보검색을 위한 Flexible matching model

다량의 오류를 포함하고 있는 문서 인식의 데이터를 효과적으로 자동 색인 하는 방법으로서 형태소 분석을 이용하는 방법과 n-gram 색인 방법을 검색 효율 측면에서 비교하였다[5]. 이 연구에서 검색 시스템으로 미국 Cornell 대학에서 개발된 SMART 시스템을 사용하였으며, 재현률(recall) 0.0부터 재현률 1.0까지 0.1단위의 11개 점에 대한 정확률(precision)을 계산하여 평균한 11-point average precision을 비교하였다. 실험 결과는 표 2.1과 같다.

이 실험의 결과 문자 인식된 데이터베이스에 대한 형태소 분석을 이용한 방법과 bi-gram을 이용한 방법 사이의 성능 차이가 0.4142와 0.4144로 거의 없는 것으로 나타나고 있다.

	형태소분석 키보드입력	bi-gram색인 키보드입력	형태소분석 문자인식	bi-gram색인 문자인식
0.1	0.8837	0.8381	0.8627	0.8093
0.2	0.8226	0.7814	0.7990	0.7581
0.3	0.5945	0.5510	0.5135	0.4998
0.4	0.5445	0.5052	0.4416	0.4412
0.5	0.5062	0.4448	0.3936	0.3965
0.6	0.4152	0.3799	0.2882	0.3178
0.7	0.3578	0.3443	0.2190	0.2776
0.8	0.2805	0.2637	0.1591	0.2058
0.9	0.2014	0.1680	0.1234	0.1346
1.0	0.1213	0.1083	0.0656	0.1075
합계	0.4930	0.4592	0.4142	0.4144

표2.1 문자 인식된 데이터베이스에 대한 색인방법의 성능비교

그러나 형태소분석을 키보드 입력과 문자인식의 결과로 보면 15% 정도의(0.4142와 0.4930의 차이) 성능 감소를 볼 수 있다. 따라서 성능의 한계가 뚜렷한 bi-gram 방법보다는 문자 인식의 성능을 개선하기 위한 오류 교정에 대한 연구가 필요하다는 점을 알 수 있다.

2.2 언어 정보를 이용한 후처리

입력 단어를 변형되지 않는 부분(head)과 활용 현상이 나타나는 부분(tail)로 구분하고, 단어가 입력되면 먼저 tail 사전을 검색하여 가능한 모든 tail을 분리한 후 나머지 부분으로 head 사전을 검색한다. head가 사전에서 발견될 경우 오류가 없는 것으로 간주하고, head가 사전에서 발견되지 않았을 경우 사전에 등록된 단어를 문자열 유사도에 기반하여 후보로 생성한 후 접속 정보를 이용하여 필터링을 한다. 이 연구의 한계로는 우선 사전에 단어가 등록이 안된 경우 교정이 불가능하고 학술 논문의 경우 전문 용어가 많아 교정 성능이 저하될 것으로 예상되며, 또한 띄어쓰기에 대한 대처방안이 제시되지 않았다는 점이다[8].

문자 인식 시스템의 인식률은 입력문자의 유사성 및 입력문자에 포함된 잡음(noise)에 의해 제약을 받게 되고 문맥적 지식(contextual knowledge)을 문자인식에 응용하여 오인식된 문자를 문법이나 의미적 지식 등으로 교정을 한 후 여러 후보 문자 중에서 문맥에 적합한 후보를 선택해서 인식기의 인식률을 높이게 한다[4]. 여기에서 언어정보는 n-gram tagging 정보와 공기패턴에 의한 상호정보(mutual information)를 사용하였다. 약 2만 2천 어절을 말뭉치(corpus)에서 자동 추출하여 실험한 결과 71.2%에서 93.53%로 인식률이 향상되었다고 제시하고 있다.

2.3 한국어 자동 띄어쓰기 관련 연구

기존의 후처리에 관한 연구는 형태소 분석기에 의존해서 언어 정보를 얻은 뒤에 교정 후보들을 수작업으로 선택함으로써 그 만큼 불필요한 오류 처리 수행이 많아지는 부담이 있다. 따라서, 띄어쓰기 문제와 문자 오인식 문제를 분리하여 먼저 띄어쓰기를 처리한 후 문자 오인식 문제를 처리하는 방향으로 한다. 이러한 문제를 해결하기 위해서 한국어 텍스트 자동 띄어쓰기에 적용하는 방식을 이용할 수 있다. 한국어 자동 띄어쓰기 처리 방식으로는 휴리스틱과 형태소 분석기를 이용한 규칙 기반 방식[3]과, 음절 통계를 이용한 통계기반 방식[9]이 있다.

규칙 기반 방식은 초등학교 교과서를 대상으로 최고 95.2%의 성공률을 보인 것으로 나타났으나, 미등록어와 신규 복합명사가 빈번하게 나타나는 텍스트에 대해서는 높은 정확률을 얻기가 힘들다. 이에 반해 통계 기반 방식은 74~85% 정도로 다소 낮은 74~85% 정도로 다소 낮은 성공률을 보이지만, 통계 데이터를 획득한 말뭉치에 그대로 적용할 경우에는 93.6%의 성공률을 보여서 통계 데이터를 얻는 말뭉치의 영향이 큰 것으로 나타났다[9].

특히 줄 경계에서의 띄어쓰기 복원을 위해 음절쌍 통계를 이용한 복원기법과 통계정보원과 음절쌍 위치에 따른 가중치를 균등하게 처리하여 높은 성공률을 보였다[7].

3. 음절 bi-gram정보를 이용한 자동 띄어쓰기

3.1 문자 인식기 오류 유형

문자 인식 오류는 인식기의 특성에 따라 다르지만 대부분 스캔 농도에 따른 획간 접촉과 끊김 현상으로 잘못 인식하는 경우가 발생하고 입력 이미지에 잡음(noise)이 들어가 인식률이 떨어지는 경우가 있다. 따라서 인식할 이미지의 해상도에 따라 민감하게 반응하며 대규모의 데이터베이스 시스템은 보통 200dpi 정도로 스캔하는데 문자인식의 성능을 고려한다면 300~400dpi로 스캔하는게 바람직하다. 본 실험에서 사용될 KT Set 논문 요약 부분을 400dpi로 스캔한 이미지이다.

HLR (Home Location Register) 가입자 정보의 관리를 위한 데이타베이스를 구축하기 위해서는 어떠한 내용의 정보들이 데이타베이스 내에 저장이 되어야 하는가를 분석 결정하여야 한다. 이러한 분석된 내용들을 데이타베이스화 하기 위한 스키마를 결정하고 이를 토대로 실제 운용될 데이타베이스가 구축된다. 실제로 데이타베이스의 구축을 위해서는 데이타 생성기라는 도구를 이용하여 구축하게 된다. 데이타베이스 생성기를 통한 데이타베이스 구축은 아래와 같이 이루어진다. 먼저 데이타베이스를 구축하기 위한 Meta-DB(Catalog Information)를 생성 하여야 한다. Meta-DB(Catalog Information)를 토대로 하여 실제 Main Memory에 구축될 DB를 구축한다. 이러한 DB의 구축을 위한 데이타 생성기 (Data Generator: DG)를 개발하여야 한다. HLR 데이타생성기(Data Generator: DG)는 초기 데이타베이스를 효율적으로 구축하여 사용자의 부담을 되도록 줄여주는 역할을 한다. 본 논문은 HLR 데이타베이스 시스템 내의 한 블럭인 초기 데이타 생성기(Data Generator)에 관한 블럭 설계서이다.

그림 3.1 실험 이미지 자료 샘플

H♣ ggsne Locadon Re♣ th우l 가입자정보의 관리를위한 태이타베이스를 구푹하기 위해서는 어떠한 내용의 정보드이 태이타때이스 내애 저장이 되어야하는가를분석 결정하여야 한다. 이러한 분석된 내룡들을 탱이타해이스화하기 위한프키마를 결정하고 이를토대로실제운응될 태이타베이스가구 푹된다. 실재로 탤이타베이스의구축을위해서는 이타 생성기라는도꾸를이용하여 후축하채 된다. 이타얘이스생성기를롱한 이타떨이스구푹은아래와 같이 이루어진다. 먼저 태이타♣ 이스를구 푹하기 위한Me이-DB♣ 80logln♣ madony- 생성 하여야한다.M6♣ DB(Cl료og Kformado마를토대로 하여 싫재 M효n Mem따,애 구축될 DB를 구축한다. 이러한 DB의 구훅을 위한 태이타 생성기 (IOta Gerer♣ t: ♣ iy. 개발하여야 한다.H♣ 데이타생성기잎aOOenaNtor DGg♣ 호기 데이타베이스를효울적으로구축하여 사응♣ 의 부당으 되도록줄여주는 역활을 한다.본 논문은H♣ : 태이타베이스사 스램 내의 한똘럭인초기 64이♣ 성기 aUGenera♣ y애 관한블럭 설♣ 서이다

그림 3.2 Speed Reader 1.2로 OCR된 결과(H730.TXT)

HI? 77ome Location Regigor) 가입자 정보의 관리를위한 데이타베이스를 구축하기 위해서는 어떠한 내용의 정보들이 데이타베이스 내애 저장이 되어야 하는가를 분석 결정하여야 한다. 이러한 분석된 내용들을 대이타태이스화 하기 위한스키마를 결정하고 이를토대로 실재 운용될 대이타베이스가구 축된다. 실제로 대이타태이스의 구축을 위해서는 대이타생성기라는도구를 이응하여 구축하개 된다.데이타베이스 생성기를 통한 대이타배이스구축은 아래와 같이 이루어진다. 먼저 대이타베이스를 구축하기 위한Meta-DB(Catalog Information)를 생성 하여야 한다. Meta-DB(Catalog Information)를토대로하여 실재 Main Memory7? 구축될 DB를 구축한다. 이러한 DB의 구축을 위한 대이타

생성기(DataGererat(r: DG)를 개발하여야 한다. HI? 대이타생성기7ata Generator DG)는 초기 대이타베이스를 효 율적으로 구축하여 사동자의 부당을 되도록 줄여주는 역활을 한다. 본 논문은B7A 대이타태이스 시스템 내의 한 블럭 인 초기 대이타 생성기7ata Generator)7? 관한블럭 설개서이다.

그림 3.3 아르미 5.0으로 OCR된 결과(H730.TXT)

상용 문자 인식기인 Speed Reader 1.2 시스템과 아르미 5.0을 이용하여 100건의 샘플 이미지를 인식한 결과 각각의 문자 인식 오류 유형 발생빈도는 표 3.1과 같다.

문자 인식기 / 오류유형	Speed Reader 1.2	아르미 5.0
문자 대치	892	864
공백 추가 / 삭제	421	413
문자 대치 + 공백 추가	59	45
문자 삭제 + 공백 추가	115	94
여러 개 기호 삽입	180	140
기 타	38	27
합계	1705	1583

표 3.1 문자 인식 오류 유형 발생빈도

문자 인식기의 오류 유형을 보면 문자 대치 오류가 약 52%, 공백 추가나 삭제 오류가 약 25%, 문자대치 및 공백 추가, 문자삭제 및 공백 추가가 약 10%정도를 차지하며 한 문자가 여러 개의 기호로 대치는 경우도 약 11%를 차지 한다. 이러한 오류의 유형은 단독으로 발생하지 않고 여러 유형이 한 단어에 동시에 발생하는 경우가 있어 오류 수정을 어렵게 만드는 요소가 된다. 그림 3.4는 문자 인식기에 따른 오류의 유형을 비교한 것이다.

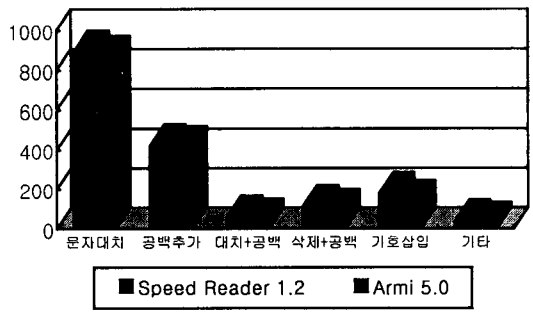

그림 3.4 문자 인식기 오류유형 비교

문자가 공백으로 대치되는 경우나 공백이 삽입되어 한 단어가 여러 개로 분리된 경우는 공백을 삭제하여 단어의 앞과 뒤에 나타나는 단어와 붙여 쓴 다음 오류를 처리하여야 한다. 그러나 "그 대로"라는 단어열의 경우 "대로"라는 단어도 가능하고 "그"라는 단어도 가능하

므로 이를 붙여서 쓴 "그대로"라는 단어와 어떤 경우가 더 옳은지 평가하여야 한다. 이를 위해 본 논문에서는 3.2절에서 단어의 확률 모델을 정의하고 확률 값이 큰 것을 선택하도록 제안한다. 단어의 확률 값을 계산하기 위한 인수들은 띄어쓰기가 잘 학습된 코퍼스로부터 자동으로 학습하여 얻을 수 있다.

3.2 자동 띄어쓰기를 위한 알고리즘

3.2.1 확률적 단어 분리 알고리즘

문자 인식된 텍스트의 경우 글자간 간격이 좁을 때나 원래의 문서에서 단어가 줄 바꿈으로 분리될 경우 띄어쓰기 오류가 많이 발생하므로 이에 대한 처리가 단어를 교정하기 전에 수행되어야 한다.

본 논문에서 제안하는 단어 인식 방법은 한국어의 음절 특성을 고려 사전을 기반하지 않고 원형 코퍼스로부터 필요한 음절 정보와 어휘 정보를 추출하는 방법을 취하므로 오류가 포함된 문서에 대하여 견고한 분석이 가능하고 많은 시간과 노력을 요구하는 사전 구축 및 관리 작업을 필요로 하지 않는다는 장점이 있다. 그러므로 띄어쓰기 오류가 빈번하게 발생하는 문자 인식 텍스트의 처리에 효과적으로 적용될 수 있다.

자동 띄어쓰기를 위한 단어 분리 알고리즘을 이용하여 음절의 열로부터 단어가 분리되는 과정을 전개하고 전개된 과정에서 단어의 확률을 구하게 된다. 그림 3.5는 자동 띄어쓰기 과정을 보인다.

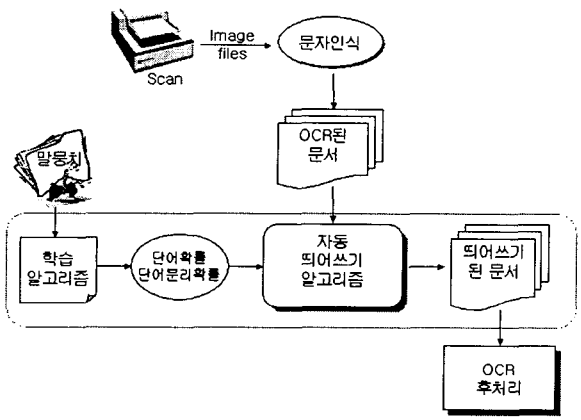

그림 3.5 자동 띄어쓰기 과정

음절의 열에서 단어를 분리하는 과정은 Luo의 연구[12]를 참조하였다.

음절의 열에서 단어를 분리하는 과정은 $s_1 s_2 ... s_n$으로 구성된 음절의 열 S에서 단어의 열 $w_1 w_2 ... w_m$을 인식하는 과정을 볼 수 있다. 이 과정을 수식으로 전개하면 다음과 같다.

$$S = s_1 s_2 ... s_n$$

$$= (s_1 ... s_{x_1})(s_{x_1+1} ... s_{x_2}) ... (s_{x_{n-1}+1} ... s_m)$$

$$= w_1 w_2 ... w_m$$

이 수식에서 x_k는 k번째 단어의 마지막 음절을 표시한다. 예를 들면 w_k는 $(s_{x_{k-1}+1} ... s_k)$의 음절로 구성된 단어이다. 이때 음절의 열 S에 대한 여러 가지 가능한 단어 열들의 후보 집합을 $G(S)$라고 한다면 $G(S)$는 다음과 같이 나타낼 수 있다.

$$G(S) = \{(x_1, x_2, ..., x_m) \mid 1 \le x_1 \le x_2 \le ... \le x_m, m \le n\}$$

위에서 정의된 개념들을 이용하여 음절의 열로부터 단어를 분리하는 과정은 음절의 열 S로부터 분리 가능한 모든 단어 열의 후보들 중 최대의 확률을 가지는 g^*를 찾는 문제로 정의하고 그 과정은 다음과 같다.

$$g^* = \arg \max_{g \in G(S)} P_g(w_1, ..., w_m)$$

위의 수식에서 $P_g(w_1, ..., w_m)$은 단어열 $w_1 w_2 ... w_m$로 분리된 후보열 g의 확률값을 나타낸다. 위의 식은 체인규칙(chain rule)에 의해 단어의 확률을 이전 단어에 대한 조건부 확률로 전개함으로써 다음과 같이 나타낼 수 있다.

$$P_g(w_1 ... w_m) = p(w_1) * p(w_2 \mid w_1) * ... * p(w_m \mid w_1 ... w_{m-1})$$

위의 수식에서 h_i는 w_i에 선행하는 단어열 $w_i ... w_{i-1}$, 즉 w_i의 history를 나타낸다. h_i를 적용하는 범위에 따라 tri-gram 모델은 $w_{i-1} w_{i-2}$를 고려하는 모델이 되고 bi-gram 모델은 w_{i-1}만을 고려하는 모델이 된다.

모델링 과정에서 많은 history 정보를 고려하는 것이 보다 정교한 단어 인식 모델을 설계하는 직접적인 방식이지만, 이 경우 모델을 학습시키기 위해서는 고려하는 history의 범위에 비례하여 많은 양의 데이터가 필요하다는 단점이 있다. 실제적으로 많은 연구들이 단어들의 uni-gram이나 bi-gram만을 history 정보로 고려하는 이유도 tri-gram 이상의 정보를 history로 고려하는 경우 학습 데이터 양과 학습 속도가 현실적인 면에서 수용하기 어렵기 때문이다. 심지어 단어의 uni-gram 정보만을 고려하는 모델도 실제 영역에 적용하는 경우 사용되는 단어 수가 매우 크기 때문에 학습에 필요한 충분한 정보를 얻지 못하는 자료희귀(data sparseness) 문제를 겪게 된다.

단어에 기반 하는 모델의 자료희귀 문제 등을 극복하

고, 띄어쓰기 오류가 빈번한 텍스트에서도 견고한 분석을 수행하기 위하여 본 연구에서는 음절 단위 정보를 이용한 단어 인식 모델을 제안한다.

3.2.2 음절 단위 정보를 이용한 띄어쓰기

음절 정보만을 이용하여 단어 인식을 올바르게 수행하기 위해서는 단어 분리에 유용한 판단 기준이 되어야 한다. 여기에서는 단어와 단어가 인접한 경계에 나타난 경우와 단어 내부에서 쓰이는 음절의 분포가 다르다는 것을 전제한다.

따라서 단어 단위의 정보를 이용하여 나타나는 자료 희귀 문제를 음절 bi-gram 정보를 이용하여 해결한다. 단어 경계와 단어 내부를 모두 고려한 확률식은 아래와 같다.

$$P(w_i \mid h_i) = (1-\lambda)p(w_i \mid w_{i-1}) + \lambda p(w_i)$$

위의 수식에서 λ는 단어 bi-gram함수와 단어 확률 함수의 값을 결합하기 위한 보정 상수(interpolation coefficient)이다.

입력 어절로부터 단어를 분리하는 과정은 정의된 단어 확률 값과 동적 프로그래밍(dynamic programming)을 이용하여 아래와 같은 알고리즘으로 나타낼 수 있다.

Initialization

$$L(i,j) = \log p(w_{oj})$$

Recursion

from i=1 to $len(w)$, j=1 to i, for all i, j

$$L(i,j) = \max_{1 \le k \le i-j} [L(j,k) + \log p(w_{ij} \mid w_{jk}) + \log p(w_{ij})]$$

$$b(i,j) = \arg\max_{1 \le k \le i-j} [L(j,k) + \log p(w_{ij} \mid w_{jk}) + \log p(w_{ij})]$$

Path Backtracking

$$i = len(w) - \max_{1 \le k \le len(w)} L[len(w)-k, k]$$

$$j = len(w) - i$$

repeat

select w_{ij}

$j = i - b(i,j)$

$i = b(i,j)$

until $i > 0$

알고리즘 3.1 동적 프로그래밍 알고리즘

4. 실험

본 절에서는 실험을 통해서 제안하는 방법이 단어 인식에 있어서 적절한 처리를 할 수 있음을 보인다. 실험에 사용하는 실험 대상으로 스캐너를 통하여 이미지를 문자 인식한 결과를 사용한다. 이 중에서 띄어쓰기가 바르지 못한 부분을 붙여 쓰기를 한 다음 bi-gram정보만으로 띄어쓰기 오류 복원을 시도하여 분절의 정확도를 확인하였다.

띄어쓰기 오류 복원은 잘못 띄어 쓴 어절들을 올바르게 복원하는 과정이다. 앞 절에서 제안하는 단어 인식과정은 한국어 문장에서 어절을 인식하는 문제에 적용할 수 있다. 예를 들어 다음과 같이 붙여 쓴 문장이 입력되었을 경우

"우리는띄어쓰기시스템을구현하였다"

단어 인식 문제는 위 문장을 다음과 같이 올바른 단어 단위로 분리하는 것이다.

"우리는 띄어쓰기 시스템을 구현하였다"

그림 4.1은 띄어쓰기가 안된 "구축을위해서는" 을 띄어쓰기가 올바르게 된 "구축을 위해서는"으로 교정하는 예를 보여준다.

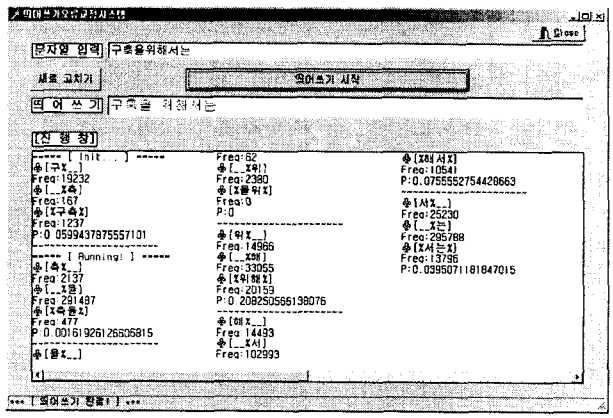

그림 4.1 띄어쓰기 오류 교정 시스템

본 실험에서는 띄어쓰기가 잘 학습된 신문기사 1년분(약 650만 어절)을 코퍼스로 이용하여 실험을 수행하였다. 붙여 쓴 어절들을 본 연구에서 제안하는 단어 인식 프로그램으로 분절한 후, 분절된 결과와 원래 띄어서 쓴 문장을 비교하여 단어 인식의 정확도를 측정하였다.

실험에 사용한 데이터는 학회 논문지 요약 부분을 400dpi로 스캔한 1,000여건 결과 중에서 두 가지 문자 인식기의 결과 100건을 가지고 띄어쓰기 오류가 발생한 부분을 전부 붙여 쓴 뒤에 띄어쓰기 복원을 해보았다. 그 결과 띄어쓰기 오류가 발생한 부분 총 1,147건중에

서 86.2%인 989건이 복원되었다. 이러한 분절의 정확도가 문자 인식 후처리 과정에 끼치는 영향은 매우 크다고 볼 수 있다.

아울러 행의 경계에서 나타나는 띄어쓰기 오류를 분석한 결과 한글은 음절 단위로 Word-Wrap 되므로 "어떠한내용의"과 같이 무조건 붙여 쓰기를 하거나 "구 축된다"와 같이 무조건 띄어쓰기를 하면 분절의 정확도가 떨어진다. 통계기반 행 경계 띄어쓰기 복원 방법과 학습된 코퍼스의 증가에 따른 분절의 정확도를 실험한 결과 저용량의 코퍼스보다는 다량의 정보를 가지는 코퍼스가 훨씬 높았다. 코퍼스의 크기의 증가에 따른 문서 내부와 행의 경계선상의 분절의 정확도는 그림 4.2와 같다.

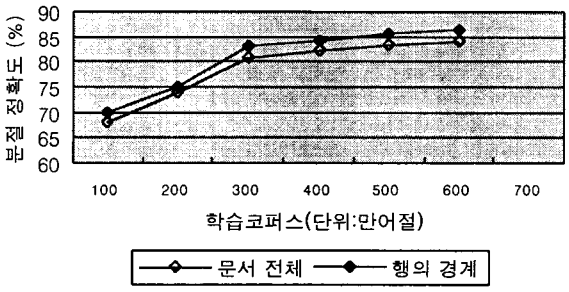

그림 4.2 학습 코퍼스에 증가에 따른 분절 정확도 비교

실제로 한국어에서는 여러 형태의 띄어쓰기 규칙이 문법적으로 허용된다. 예를 들면 "정보검색시스템"과 " 정보 검색 시스템" 혹은 "정보검색 시스템"이 모두 올바른 띄어쓰기로 볼 수 있다. 본 실험의 경우는 위와 같이 문법적으로 허용하는 올바른 띄어쓰기이더라도 원래의 문장과 다르면 오류로 처리하였기 때문에 복합명사에 의한 오류가 다수 발생하였다. 이 밖에도 띄어쓰기 오류 회복의 문제점은 학습코퍼스의 크기에 따라 분절의 정확도가 달라지고 전문 용어나 고어/신조어의 처리를 위한 새로운 코퍼스가 필요하며 속도면에서는 사전을 이용하는 것보다는 효율성이 떨어짐을 볼 수 있었다.

5. 결론 및 향후 연구방향

문자인식기가 가지는 띄어쓰기 오류를 자동으로 수정하기 위해서 문자 단위가 아닌 어절 단위의 오류 수정 기법을 이용하여 자동 띄어쓰기의 모델을 제시하였고 보통의 띄어쓰기 오류는 공백의 추가에 의하거나 삭제에 의해 발생하므로 학습이 잘된 코퍼스를 이용하여 띄어쓰기 오류 회복의 효율성을 높였다.

그러나, 실험에서 띄어쓰기 복원에 실패한 경우는 크게 학습 코퍼스의 크기에 따라 분절의 정확도가 달라져서 오류가 발생하였고 다른 하나는 미등록어 및 신조어를 포함하는 단어가 나타나면 문자 인식기의 오류로 판단하는 경우가 발생하게 된다. 이를 위해서 미등록어 및 신조어를 판단하고 이를 처리할 수 있는 모델이나 최신의 코퍼스에 대한 연구가 필요하다. 또한 통계기반 방식과 언어 규칙기반 방식을 혼용하는 시스템에 대해서도 연구가 필요하다.

6. 참고문헌

[1] 이현민, 전남열, 박혁로, "문자 인식된 데이터베이스 자동 색인을 위한 오류 회복 기법 연구", 연구개발정보센터, 1999.

[2] 한선화, 이충식, 이준호, 김진형, "문자 인식 기술을 데이터베이스 구축에 활용하기 위한 사전 연구," 한국정보처리학회 논문지, 1999.

[3] 신중호, 박혁로. "음절단위 bigram정보를 이용한 한국어 단어인식모델," 제9회 한글 및 한국어 정보처리 학술대회 발표논문집, pp.255-260. 1997.

[4] 유진희, 이종혁, 이근배, "형태소 분석과 언어평가를 이용한 문자 인식 후처리", 정보과학회 논문지 제22권 제6호, 1995.

[5] 이준호, 이충식, 한선화, 김진형, "문자 인식에 의해 구축된 한글 문서 데이터베이스 정보 검색", 정보처리학회지, 제6권 제4호, 1999.

[6] 손훈석, 최성필, 권혁철, "문자 인식기의 특성과 말뭉치의 통계 정보를 이용한 문자 인식 결과의 후처리", 한글 및 한국어 정보처리 학술발표논문집, 1997.

[7] 정영미, 이재윤. "한국어 텍스트 처리를 위한 줄 경계 띄어쓰기 복원", 제6회 한국정보관리학회 학술대회 논문집, 21-24, 1999.

[8] 이병희, 김태균, "오프라인 한글 문자 인식을 위한 효율적인 오인식 단어 교정 방법," 한국정보처리학회 논문지, 제3권 제6호, 1998.

[9] 심광섭, "음절간 상호정보를 이용한 한국어 자동 띄어쓰기," 정보과학회논문지(B), 23(9): pp.991-1000. 1996.

[10] 김성혁 외 5인, "자동 색인기 성능 시험을 위한 Test Set 개발," 정보관리학회지 제 11권 제 1호, pp. 81-101, 1994.

[11] 이성환, 문자 인식 이론과 실제, 홍릉과학출판사, 서울, 1994.

[12] Luo, Xiaogiang, Salim Roukos, "An Iterative Algorithm to Build Chinese Language Models,", Proceedings of 34th Annual Meeting of the Association for Computational Linguistics, 1994.

[13] Chaniak. E., Statistical Language Learning, MIT Press, Cambridge, MA, 1993.