음성 자료에 대한 규칙 기반 Named Entity 인식

김지환(LG전자기술원)

<차 례>

<Abstract>

Rule-based Named Entity (NE) Recognition from Speech

Ji-Hwan Kim

In this paper, a rule-based (transformation-based) NE recognition system is proposed. This system uses Brill's rule inference approach. The performance of the rule-based system and IdentiFinder, one of most successful stochastic systems, are compared. In the baseline case (no punctuation and no capitalisation), both systems show almost equal performance. They also have similar performance in the case of additional information such as punctuation, capitalisation and name lists. The performances of both systems degrade linearly with the number of speech recognition errors, and their rates of degradation are almost equal. These results show that automatic rule inference is a viable alternative to the HMM-based approach to NE recognition, but it retains the advantages of a rule-based approach.

* Keywords: Named entity recognition, Transformation-based rule inference, Automatic rule generation. Speech recognition.

1. 서 론

음성으로부터의 정보 추출(information extraction)은 음성인식(speech recognition) 기술이 언어이해(speech understanding)의 수준으로 발전하는데 있어서 가장 중요한 단계들 중 하나이다. 현재 음성을 포함하는 멀티미디어 자료에 대한 효과적인 검색 방법에 대한 연구가 활발히 진행 중이며, 특히 검색에서 많이 사용되는 Named Entity(NE)의 멀티미디어 자료로부터의 인식은 음성인식 기술의 언어이해 수준으로의 발전에 가장 중요한 부분 중 하나가 되고 있다.

NE 인식은 6차 Message Understanding Conference에서 지명, 인명, 기관명과 같은 고유명사, 날짜와 시간과 같은 시간정보, 금액과 백분율과 같은 숫자 표현들을 인식하는 것으로 정의되었다[1].

NE의 많은 수가 고유명사이기 때문에 대소문자 정보와 구두점 정보는 NE 인식에 있어서 중요한 정보가 된다. 따라서 대소문자 정보와 구두점 정보가 제공되는 일반 텍스트에서는 대소문자 정보와 구두점 정보를 이용하여, NE 인식기를 상대적으로 쉽게 구현할 수 있다.

그러나, 일반 멀티미디어 자료에서는 대소문자 정보와 구두점 정보가 일반적으로 제공되지 않는다. 화자가 자신의 음성이 음성인식이 된다는 사실을 아는 경우에는 "쉼표", "마침표", 또는 "현재의 단어를 대문자로" 등과 같은 명령어를 화자가 발성함으로써 대소문자 정보와 구두점 정보를 생성할 수 있지만, 방송 자료에 대한 음성인식과 같이 화자가 자신의 음성이 음성인식이 된다는 사실을 모르는 경우, 화자에게 대소문자 정보와 구두점 생성을 위해 필요한 명령어 발성을 요구할 수 없게 되기 때문이다.

대소문자 정보와 구두점 정보가 제공되지 않는 것 외에도, 멀티미디어 자료에 대해서 음성인식이 수행되는 경우 발생하는 음성인식 오류도 멀티미디어 자료에 대한 NE 인식을 더욱 어렵게 만든다. 음성인식 오류에 의한 입력 자료의 오염은, NE 인식을 위해 학습된 패턴을 구성하는 엘리먼트들의 정합을 방해하기 때문이다.

본 논문에서는 Brill의 변환 기반 규칙 추정 방식을 이용하여 규칙을 자동으로 생성하는 규칙 기반의 NE 인식 방법을 제안하며, 제안한 방법을 뉴스 음성 자료에 대한 NE 인식에 적용한다. 2장에서는 기존의 연구 방법이 기술된다. 3장에서는 Brill의 변환 기반 규칙 추론 방법이 소개되며, 이를 이용한 규칙을 자동으로 생성하는 규칙 기반의 NE 인식기 구현 방법을 기술한다. 그 후 4장에서 실험 방법과 실험 결과가 설명된다. 마지막으로 5장에서 본 논문의 내용과 결과를 정리한다.

2. 기존의 연구

NE 인식 분야는 역사가 길지 않기 때문에 현 시점에서 NE 인식에만 초점을 맞춘 저널지나 책이 있는 것은 아니다. 학회 또는 워크샵으로는 Message Understanding Conference(MUC)[2][3]와 DARPA Broadcast News(BN) Workshop [4]이 있었고, 이들 학회지에는 해당 학회와 워크샵에서 수행한 성능평가에 참여한 시스템과 성능평가 결과에 대해서 기술이 되어 있다. 기존의 연구는 MUC와 DARPA BN Workshop을 위주로 설명을 한다.

MUC에서는 도메인이 한정된 텍스트 자료에 대해서 평가가 수행되었다. 도메인이 한정된 텍스트 자료이기 때문에 특히 대소문자 정보가 NE를 인식하는데 결정적인 역할을 했었고, 따라서 MUC에 참여한 대부분의 시스템들은 수동으로 작성한 규칙에 기반한 시스템들이었다. MUC에 참여한 규칙 기반 NE 인식 시스템들의 대표적인 예들은 [5][6][7][8]에 기술되어 있다.

이에 반해 DARPA BN Workshop에서는 NE 인식이 뉴스에 대한 음성 자료에 대해서 수행되었기 때문에 뉴스가 다루는 많은 도메인을 처리할 수 있어야 하고, 또한 대소문자 정보와 구두점 정보가 없는 경우에 대해서도 NE 인식이 수행되어야 했다. DARPA BN Workshop에 참여한 시스템들은 통계적 방법을 채택한 경우[9][10][11]와 수동으로 작성한 규칙에 기반한 방법을 채택한 경우[11][12]로 나뉘어진다.

따라서, MUC와 DARPA BN Workshop에서 수행한 성능평가에 참여한 시스템들을 위주로 기존의 NE 인식 시스템들을 구분해 보면 크게 HMM기반의 통계적(stochastic) 방식과 수동으로 작성한 규칙 기반의 방식으로 나뉘어진다[13].

규칙 기반의 방식에서는 언어 정보가 단순한 형태의 규칙으로서 직접적으로 기술되게 된다. 따라서, 적은양의 메모리 요구량을 가지게 되며, 통계적 방식에서 사용되는 back-off[14] 모델과 같이 less-descriptive한 모델을 요구하지도 않으며, 개념적으로 쉽게 이해되는 규칙들로 이루어지기에 언어에 대한 지식이 쉽게 규칙으로 기술되어 확장될 수 있는 장점을 가지고 있다. 그러나, 기존의 규칙 기반의 방법들은 규칙을 수동으로 만들어야 하는 단점을 가지고 있다[13].

텍스트 자료에 대한 NE 인식의 경우, 수동으로 만들어진 규칙들을 사용하는 규칙 기반의 방법들은 좋은 NE 인식율을 보였는데, 이는 일반적인 텍스트 자료는 대소문자 구분이 되어 있기 때문이며, 따라서 대소문자 정보에 관련된 적은수의 규칙만으로도 상당수의 NE를 추출할 수 있기 때문이다[13]. 그러나, 만일 NE 인식이 음성 자료에 대해서 적용되는 경우, 대소문자 정보가 없기 때문에 필요한 규칙들을 모두 수동으로 만드는 것은 매우 힘들어 지게 되며, 이로 인해 시스템을 개발하는 전체 시간이 길어지게 된다.

규칙 기반의 방법과는 달리 통계적 방법에서 언어 정보는 통계 자료 처리를

위한 큰 테이블의 형태로 간접적으로 나타나지게 된다. 이 방법은 전 과정이 자동화되어 있어 시스템을 개발하는 전체 시간이 크게 줄어드는 장점을 가지고 있다. 그러나, 통계 자료 처리를 위한 충분한 데이터를 얻는데 많은 양의 학습자료가 필요하게 되는 단점을 가지게 된다.

통계적 방법에서 많이 사용되는 모델은 Hidden Markov Model (HMM)[15][16][17]이며, DARPA BN Workshop에서 가장 좋은 성능을 보인 BBN의 IdentiFinder 시스템도 HMM을 이용하고 있다. 통계적 방법을 이용한 NE 인식에서는 HMM의 상태 (state)에 NE의 클래스를 대응 시킨다. 상태간 천이 확률 (transition probability)은 이전 NE 클래스가 주어진 경우 현재의 NE 클래스로 이동하는 확률이며, 출력 확률 (output probability)은 현재의 NE 클래스에서 해당 단어가 사용되는 확률을 뜻하게 된다.

3. 규칙 기반 Named Entity 인식

기존의 규칙 기반의 방법들이 가지고 있는 한 가지 단점은 규칙들을 작성하는데 많은 노력이 필요했다는 점이다[18]. 수동으로 규칙들을 작성해야 했기에, 만들어진 시스템을 새로운 언어나 새로운 도메인에 적용하기 위해서 필요한 모든 정보를 수동으로 작성하는 것은 매우 힘든 일이기 때문이다.

언어 정보를 코퍼스로부터 자동으로 추출하는 시스템은 다음의 두 가지 장점을 가진다. 첫째, 총 개발 시간이 크게 단축된다. 둘째, 코퍼스 분석에 기반한 시스템은 통계적인 특성들을 코퍼스로부터 학습하기 때문에 학습자료에 지나치게 튜닝이 되는 문제를 피할 수 있게 된다[19].

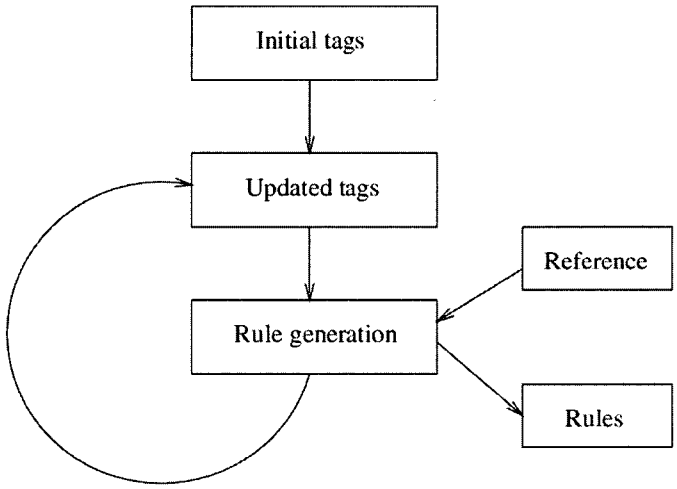
Brill은 코퍼스로 부터 규칙을 추출한 후, 추출된 규칙에 기반한 형태소 태거를 개발하였다[19][20][21]. 그의 연구에서, 학습의 과정은 태깅이 전혀 되어 있지 않은 텍스트로부터 시작된다. 학습의 각 단계에서, 학습 프로그램은 텍스트에 적용시 태깅 성능에 가장 좋은 향상을 가져오는 변환규칙을 발견한다.

각 규칙에 의한 성능향상 정도는 답안 태그와 규칙이 적용된 후 태그들을 비교함으로써 계산되어진다. 이것이 확률 기반의 방법과 변환에 기반한 방법과의 가장 중요한 차이점이다. 확률기반의 방법은 입력에 대한 생성 확률이 최대가 되도록 학습이 진행되지만[1], 변환에 기반한 방법은 에러의 개수를 최소화 하도록 학습이 진행된다. 태깅 성능에 가장 좋은 향상을 가져오는 규칙이 결정된 다음 해당 규칙은 저장되고, 현재의 태그들을 변환시키기 위해서 본 규칙이 적용된다. 이 과정은 더 이상 태깅 성능의 향상을 가져오는 변환 규칙이 발생되지 않을 때까지

1) Maximum Likelihood (ML) 학습으로 가정함

계속된다.

<그림 1>은 본 학습 과정을 나타낸다.



<그림 1> 규칙 기반 방식의 학습 과정

변환 기반 방식을 적용하기 위해서는 다음의 사항이 정의되어 있어야 한다.

1. 초기 태깅값 구하는 방법 (preprocessing)
2. 매 단계 변환을 확인하는 규칙 생성 엔진
3. 정답 태그들과 현 태그들을 비교해주며, 가장 좋은 변환을 선택 가능하게 해주는 스코어링 함수

태깅 정확도가 Brill의 연구에서는 스코어링 함수로 사용되었다. 규칙들은 규칙 생성 단계가 반복될 때마다 규칙 템플레이트에 의해서 생성된다. Brill 형태소 분석기에서는 21개의 규칙 템플레이트가 사용되었다[21]. 규칙 템플레이트의 예는 [21]에 다음과 같이 나타나 있다.

다음 경우에 i번째 단어의 형태소 z_i를 태그 z_i'로 변경한다.

이전의 (다음의) 단어가 z로 태그된 경우
이전의 (다음의) 단어가 w인 경우
두개 이전 (이후) 단어가 w인 경우
하나 또는 두개 이전 (이후) 단어가 z로 태깅된 경우
현재의 단어가 w이고, 직전 (직후) 단어가 w'인 경우
현재의 단어가 w이고, 직전 (직후) 단어가 z로 태깅된 경우

형태소 태깅을 위해서 생성된 한 가지 규칙의 예는 다음과 같다.

"만약 직전 단어의 형태소가 한정사 (determiner)이면, 현재 단어의 형태소 태그를 동사에서 명사로 변경한다."

순서에 따라 생성된 변환규칙들이 학습된 이후, 새로운 텍스트는 각각의 변환규칙들을 순서에 따라 적용함으로써 태깅이 된다.

Brill의 변환기반 형태소 태거와 확률기반 형태소 태거의 성능이 [20]에서 비교되었다. Wall Street Journal 코퍼스로부터 학습된 확률기반의 형태소 태거의 결과는 [22]으로부터 인용되었다. 같은 조건에서 테스트하기 위해서 Brill의 형태소 태거는 같은 자료를 이용해서 학습 및 평가가 되었다. 이 비교에서, 변환 기반 방식에서는 비록 단지 267개의 단순한 규칙들만이 학습되었지만, 10,000여개의 확률들이 학습된 확률 기반의 방식에 비해서 변환기반 형태소 태거가 더 좋은 성능을 보였다.

3.1.장에서 설명될 규칙 기반의 NE 인식기 시스템의 기본 개념은 Brill의 형태소 태거에서 출발한다. 몇몇 NE 인식시스템은 Brill의 형태소 태거를 단지 그들의 NE 인식시스템의 전처리기로만 사용을 했다[23][24]. 그러나, 본 논문의 NE 인식 시스템에서는 Brill 형태소 태거에서 사용된 아이디어를 바탕으로 구현되었다: 즉, 모든 NE 인식 규칙은 Brill의 아이디어를 사용해서 자동으로 생성된다.

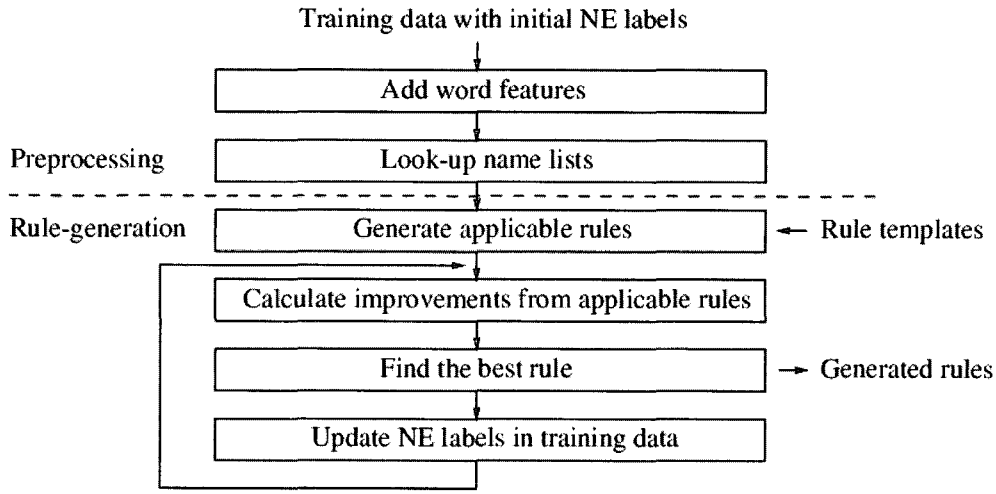
3.1. 변환 기반 자동 Named Entity 규칙 생성

<그림 2>는 제안한 규칙 기반 자동 NE 규칙 생성 시스템에서의 단계들을 보여준다. 단계들은 크게 전처리 부분과 규칙 자동 생성의 두 부분으로 나누어진다. 전처리 (preprocessing) 부분은 3.1.1.에서 기술된다. 그 후 Brill의 형태소 태거[19]에서 기본 아이디어를 얻은 자동 규칙 생성 부분단계에 대해서 3.1.2.에서 서술한다.

3.1.1. 전처리 (preprocessing)

저장 공간 요구량과 단어 길이의 불균일성 때문에, 단어들을 문자열로 메모리나 디스크에 저장하는 것은 효율적이지 않다. 학습자료 내의 모든 단어들은 모든 단어들이 대문자화 되어 단어 리스트에서의 index로 변환된다. Index 0, 1 ,2는 특수어 (각각 문장 시작 (+START+), 문장 끝 (+END+), 단어리스트 미등록어 (+UNKNOWN+))를 위해서 사용된다. ' and 'S과 같은 소유격의 단어들이 NE 단어들과 결합되는 경우, 인식기는 소유격 단어들을 NE 단어들로 부터 분리시킨다 (예: <*ENAMEX TYPE="ORGANIZATION"*> NASDAQ <*/ENAMEX*>'S). 따라서, 시스

템이 단어 리스트를 만들려고 할 때, 모든 소유격 단어들은 분리되고 별도의 단어들로 처리된다.



<그림 2> 규칙 기반 자동 NE 규칙 생성 시스템에서의 단계들

문장의 문법적 구조는 부분적으로 쉼표와 마침표와 같은 구두점에 의해서 나타난다. 따라서, 모든 구두점들이 텍스트와 함께 제공된다면, 그 시스템의 성능은 향상될 것이다. 이 논문에서 개발된 시스템은 모든 구두점들을 인접한 단어들과 분리시키고, 이들 구두점들을 별도의 단어로 취급한다. <그림 3>은 단어들을 인덱스로 변환하는 예를 보여준다.

```
W    +Start+ Wages  in    the  United States have gone  up   only about three
IW      0    23333 10682 21629 22700 20488 9790 9205 22844 14856  70   21748
W    and a half percent  in    the  past  year  ,  while global competition is
IW   844 14 9593 15668 10682 21629 15472 24038 11 23657 9139   4275   11338
W    one  reason for  the  slow growth in  pay  . +End+
IW   14847 17372 8473 21629 19837 9453 10682 15552 12  1
```

<그림 3> 단어에서 단어리스트의 인덱스 변환 예 (W: 단어; IW: 단어리스트에서의 단어 인덱스)

몇몇 NE들은 한개 이상의 단어로 구성되어지기 때문에, 주변 단어와 연결되어 NE 단어를 형성하는 NE 경계 정보를 제공, 유지하는 것은 NE 인식기 시스템의 구현에 있어서 매우 중요하다. 예를 들어, "Tony"와 "Blair"의 NE 클래스가 같지만,

<ENAMEX TYPE="PERSON"> Tony </ENAMEX>
<ENAMEX TYPE="PERSON"> Blair </ENAMEX>

와

<ENAMEX TYPE="PERSON"> Tony Blair </ENAMEX>

은 다르게 된다. 구현에 있어서는, NE 경계를 나타내기 위해서 저장 공간이 할당된다. 각각의 저장 공간에는 초기 값으로 0이 저장된다. 그 후, 만약 현재의 단어가 그 앞의 단어와 연결되어 하나의 NE 단어를 형성하게 되면, NE 경계 정보를 위한 기억 공간에 값은 1로 바뀌게 된다.

단어특성(word feature)은 때때로 NE 인식에 대한 좋은 단서를 제공한다[18][25]. 예를 들어, 단어의 첫 번째 문자가 대문자이고, 그 단어가 문장의 첫 번째 단어가 아니라는 것은 고유명사인 NE 단어일 가능성이 더 높다는 것을 나타낸다. <표 1>은 단어특성의 예들을 보여주고 있다. 처음 두개의 단어특성들 (Fst_Cap, All_Cap)은 해당 단어의 문자들이 대문자로 되어 있는지의 여부로써 판별된다. 그 다음 세 개의 단어 특성들 (Not_in_Ent, Ent_in_L, Ent_in_R)은 NE 단어에 대한 NE가 아닌 단어들의 관계를 관찰하는데 사용된다. 이러한 특성들은 단어 리스트가 만들어질 때 작성된 테이블을 참조함으로써 얻어지게 된다.

마지막 특성인 NUMERIC은 숫자와 시간 엔터티들을 구분하기 위해 사용된다. 이러한 특성들은 numeric 사전을 검색하는 것으로부터 얻을 수 있는데, 이러한 numeric 사전은 수동으로 만들어진다. 본 시스템은 63개의 단어를 가진 numeric 사전을 사용한다.

<표 1> 단어특성 (word feature)

타입	설명
Fst_Cap	문장의 첫 단어는 아니면서 첫 번째 문자가 대문자인 단어
All_Cap	두문자 이상으로 이루어진 단어로 모든 문자가 대문자인 단어 (예: NASDAQ)
Not_in_Ent	NE 내부에서 사용된 적이 전혀 없는 단어
Ent_in_L	Not_in_Ent이고 좌측에 NE를 가질수 있는 가능성이 있는 단어
Ent_in_R	Not_in_Ent이고 우측에 NE를 가질 수 있는 가능성이 있는 단어
NUMERIC	numeric 사전에 있는 numeric 단어들

단어 특성들은 서로 간에 배타적인 것이 아니기 때문에, 하나의 단어는 한개보다 많은 단어 특성들을 가질 수 있다.

NE 인식에 있어서 코퍼스 기반 방식의 근본적인 제약은 아주 큰 학습자료에서조차도 상대적으로 매우 적은 숫자의 엔터티들 (인명, 지명, 기관명 등)만이 나타

난다는 것이다[13]. 미등록어 모델을 사용했을 경우에도, 이러한 엔터티들의 식별은 시그날링 단어 (예: Mr.)의 존재로써 대부분 이루어지게 된다. 한가지의 해결방법은 지명, 이름, 잘 사용되는 성, 기관명 등에 대한 리스트를 사용하는 것이다. 이 방법의 장점은 많은 수의 엔터티들을 쉽게 포함시킬 수 있다는 것이다. 만일 같은 수의 엔터티들을 일반적인 텍스트로부터 얻는다면 엄청난 규모의 코퍼스가 필요하게 된다.

NE 타입별 네임리스트를 이용하여 NE에 대한 초기값을 주는 것은 가능하지만, 이러한 근거만으로 NE 타입을 확정하는 것은 바람직하지 않다. 예를 들어 "Berlin"의 경우, 자료에서 많은 경우 지명으로 사용되고 지명 리스트에 나타나겠지만, "Berlin Orchestra"의 경우에서와 같이 기관명으로 사용되는 경우를 배제해서는 안된다.

이름과 장소명에 대한 네임리스트에서 문맥상의 정보는 거의 제공되지 않는다. 그러나, 기관명 리스트에서 엔터티는 문맥상 정보로 사용될 수 있는 복수개의 단어들과 결합되게 된다. 이 경우, 엔터티들은 반복적으로 of나 the를 포함하게 되는데, 이들이 규칙 또는 언어모델로서 반영이 되게 되면, 리스트에서 많은 수의 of와 the가 발견되므로, 이들이 규칙과 언어모델의 왜곡을 가져와 입력 텍스트의 많은 부분의 of와 the에서 잘못된 태그값을 가지게 된다.

본 시스템에서 네임리스트로부터 얻은 단어 특성은 전처리 단계에서 단어 특성으로서 첨가가 된다. <표 2>는 네임리스트로부터 도출된 단어 특성들을 나타내고 있다. 인명, 장소명, 기관명 네임리스트들이 사용되었다. 규칙 기반 시스템이 이러한 정보들을 결합할 때 만약 한 개 이상의 단어 리스트의 엔터티가 중복된다면, 이 시스템은 긴 엔터티를 선호한다. 만약 같은 단어가 한개 이상의 네임리스트에서 나타나진다면, 우선순위가 적용된다. 장소명 네임리스트가 가장 높은 우선순위를 가지게 되고, 인명 네임리스트가 그 다음 우선순위를 가지게 되고, 기관명 네임리스트가 가장 낮은 우선순위를 가지게 된다.

<표 2> 네임리스트로부터 구해진 단어 특성들

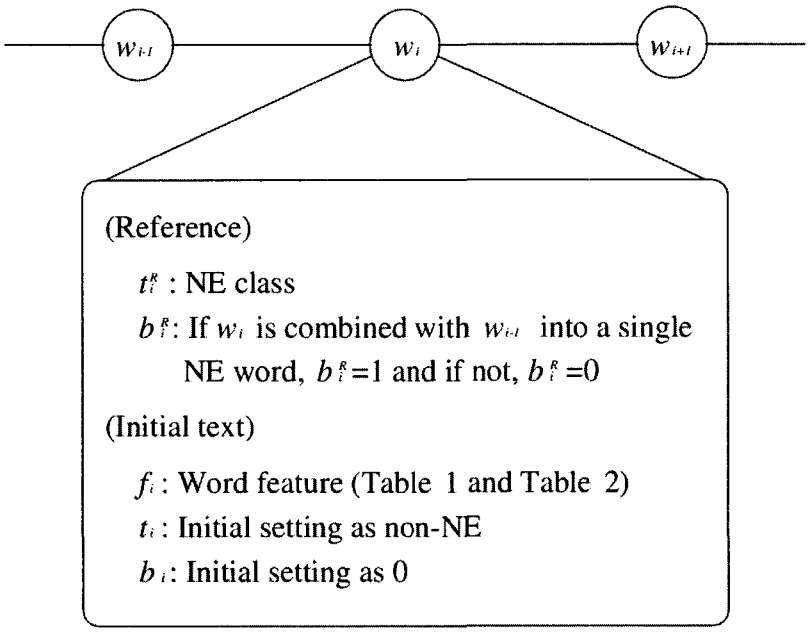
타입	설명
In_P_List	인명 네임리스트에 있는 단어
In_L_List	장소명 네임리스트에 있는 단어
In_O_List	기관명 네임리스트에 있는 단어

<그림 4>는 전처리 단계의 결과를 요약해서 보여주고 있다. w_i는 i위치의 단어를 나타낸다. 3.1.2.에서 기술될 규칙 생성의 단계에서, 규칙들은 답안 텍스트의 NE 타입 및 그 경계와, 입력 텍스트의 NE 타입 및 그들의 경계를 비교함으로써 생성된다. 이 비교를 수행하기 위해서 w_i의 답안 NE 타입은 t^R_i로 저장된다. b^R_i는 답안 텍스트에서 w_i가 w_{i-1}과 결합되어 하나의 NE 단어를 만드는지를 (예: Labour

Party) 나타낸다. 즉, w_i가 결합되어 있다면, b^R_i=1이고 그렇지 않다면, b^R_i=0가 된다.

전처리 과정에서 NE 태그의 초기값들이 주어지게 된다. f_i는 w_i의 word feature를 뜻하는데, 이 값은 w_i의 특성에 따라, 또는 네임 리스트를 검색함으로써 구해진다. 단어 특성의 세부내용은 <표 1>에 주어져 있고 사용되어진 네임 리스트는 <표 2>에 열거되어 있다.

현재 텍스트에 대한 w_i, f_i, t_i, b_i 값들에 기반해서 t_i, b_i와 답안 텍스트의 t^R_i, b^R_i 간의 차이를 줄이는 방향으로 적용 가능한 규칙들이 생성된다. t_i의 초기값은 non-NE로 주어지며, b_i의 초기값은 0으로 주어져 있다. 규칙생성의 세부적인 내용은 3.1.2.에서 서술한다.



<그림 4> 변환 기반 자동 NE 규칙 생성의 전처리 단계

3.1.2. 규칙 생성과 테스트

전처리 단계가 끝난 후, 자동 규칙 생성 과정이 수행된다. 규칙 생성 과정에서는 현 텍스트의 NE 타입과 NE 경계가 답안 텍스트의 NE 타입과 NE 경계와 비교가 되며, 그 후 에러들의 숫자가 세어지게 된다. NE 타입과 NE 경계가 정확하지 않은 모든 단어들에 대해서, NE 타입과 NE 경계를 정확하게 만들어주는 규칙들이 생성되고, 저장되며, 그리고 적용이 되며, 학습 텍스트 자료 전체에 대해서 개선을 가져오는 숫자들이 세어지게 된다.

규칙들은 규칙 템플레이트에 의해서 생성되어진다. <표 3>은 이 시스템에서 사

용된 53개의 규칙 템플레이트를 보여주고 있다. 규칙 템플레이트는 문자들과 아래 첨자로 구성되어 있다. *w*, *f*, *t*는 템플레이트들이 단어와, 단어 특성, NE 타입에 각각에 연관되어 있음을 뜻한다. *b*는 해당 단어가 이전 단어와 결합이 되어 하나의 NE 단어들 구성하는지를 보여준다 (연결되어 있다면 *b*=1, 연결되어 있지 않다면 *b*=0). 아래 첨자는 현재 단어로부터의 상대적 거리를 나타낸다. 즉, 0은 현재 단어를 뜻하고, -1은 하나 앞 단어를 뜻하며, 1은 다음 단어를 뜻한다.

<표 3> 53개 규칙 템플레이트와 해당 단계 번호 (w:단어; f:단어 특성; t:NE 타입). 아래첨자는 현재 단어와의 거리를 나타내며, 괄호안의 숫자는 규칙 적용 범위를 나타낸다[현재 단어로부터 시작점의 오프셋, 현재 단어로부터 종료지점의 오프셋]

단계 번호	규칙과 적용 범위
0	w0 f0 [0 0], w0 f-1 [-1 0], w0 f1 [0 1]
1	w0 w1 [0 1], w0 w-1 [-1 0], w0 t1 [0 1], w0 t-1 [-1 0], w1 t0 [0 1],
	w-1 t0 [-1 0], t0 t1 [0 1], t0 t-1 [-1 0], w0 f-1 [-1 0], w0 f1 [0 1]
2	w0 w-1 w-2 [-2 0], w0 w1 w2 [0 2], w0 w-1 w1 [-1 1], w0 t1 [0 1],
	w0 t-1 [-1 0], w1 t0 [0 1], w-1 t0 [-1 0], w0 w1 t2 [0 2],
	w0 w1 t-1 [-1 1], w0 t1 w2 [0 2]
3	w0 f0 b0 [0 0], w0 f0 b0 b1 [0 0]
4	w0 w-1 t0 t-1 [0 0], w0 w1 t0 t1 [0 0]
5	w0 f0 [0 0]
6	w0 t0 t-1 [-1 0], w0 t0 t1 [0 1]
7	w-1 w-2 t0 f0 [0 0], w1 w2 t0 [0 0], w-1 t0 [0 0], w1 t0 [0 0]
8	w-1 f-1 f0 [-1 0], w1 f1 f0 [0 1], w0 f0 t-1 [-1 0], w0 f0 t1 [0 1]
9	w-1 f-1 f0 [0 0], w1 f1 f0 [0 0], w0 f0 t-1 [0 0], w0 f0 t1 [0 0]
10	w0 t-1 t1 f0 [-1 1], w0 f-1 f1 f0 [-1 1], w0 f1 w2 [0 0], w0 f-1 w-2 [0 0],
	w0 f1 t2 [0 0], w0 f-1 t-2 [0 0]
11	w0 w-1 [0 0], w0 w1 [0 0], w0 w-1 w-2 [0 0], w0 w1 w2 [0 0],
	w0 w-1 w1 [0 0]

각각의 규칙 템플레이트는 규칙 적용 범위가 있는데, 규칙의 조건이 맞는 경우, 규칙에 의해서 NE 타입의 변화가 생기는 범위를 뜻한다. 예를 들어, 생성된 다음의 규칙, '만일 w_0=DOLLARS이고 f_{-1}= NUMERIC이면 NE 타입을 MONEY로 바꿀 것'에 대해서 생각해 보자. 이 규칙은 규칙 템플레이트 w_0 f_{-1}과 범위 [-1 0]에 의해서 생성되어지며, 만약 현재의 단어가 'DOLLARS'이고 이전단어의 단어 특성이 'NUMERIC' 이면 현재 단어와 이전 단어의 NE 클래스를 'MONEY'로 변경한다는 뜻이다. 그 후, 이전 단어와 현재 단어를 *<NUMEX TYPE="MONEY">* five dollars *</NUMEX>*에서와 같이 하나의 NE 단어로 결합한다.

각각의 가능한 규칙에 대한 향상 정도는 한 개의 규칙이 생성되는 경우마다 매회 업데이트 된다. 만약 모든 53개의 규칙 템플레이트가 동시에 사용되어진다면, 이러한 업데이트에 필요한 계산량은 매우 커지게 된다. 요구 계산량을 줄이기

위해서, 규칙 템플레이트는 12개의 세트로 그룹핑이 되며, 각각의 규칙 생성 단계는 이들 규칙 템플레이트 세트에 의하여 나누어지게 된다.

각각의 단계에서 생성 가능한 규칙들로부터 가장 많은 향상정도를 나타내는 규칙이 현재의 학습 텍스트에 적용되고, 학습 텍스트가 업데이트 된다. 학습 텍스트의 업데이트 이후, 다른 규칙에 영향을 주는 NE 타입과 NE 경계에 어떠한 변화가 생기게 되는 경우, 이를 다른 규칙의 향상정도 계산에 반영한다. 이 시스템에서 규칙의 적용시 향상정도는 그 규칙이 적용된 후, 텍스트의 NE 타입과 NE 경계를 정확하게 만들어 주는 숫자로 정의한다. 이 과정은 현 학습 텍스트의 NE 타입 및 NE 경계와, 답안 NE 타입과 NE 경계간의 에러의 숫자를 줄여주는 규칙이 발견되지 않을 때까지 계속 반복된다. <표 4>는 학습과정이 시작되는 시점에서 가장 큰 향상정도를 보여주는 6개의 규칙이다.

<표 4> 학습 시작시 가장 많은 향상정도를 보이는 6개의 규칙과 해당 규칙 템플레이트

규칙	템플레이트
현 단어가 'DOLLARS'이고, 이전 단어의 특성이 'NUMERIC'인 경우, 현재 단어와 이전 단어의 NE 타입을 'MONEY'로 변경	w0 f-1 [-1 0]
현 단어가 'NINETEEN'이고, 현 단어 특성이 'NUMERIC'인 경우, 현 단어의 NE 타입을 'DATE'로 변경	w0 f0 [0 0]
현 단어가 'PERCENT'이고, 이전 단어의 특성이 'NUMERIC'인 경우, 현 단어와 이전 단어의 NE 타입을 'PERCENT'로 변경	w0 f-1 [-1 0]
현 단어가 'DOLLAR'이고, 이전 단어의 특성이 'NUMERIC'인 경우, 현 단어와 이전 단어의 NE 타입을 'MONEY'로 변경	w0 f-1 [-1 0]
현 단어가 'CLINTON'이고, 현 단어의 첫 번째 문자가 대문자로 된 경우, 현 단어의 NE 타입을 'PERSON'으로 변경	w0 f0 [0 0]
현 단어가 'HOUSE'이고, 현 단어의 첫 번째 문자가 대문자인 경우, 현 단어의 NE 타입을 'ORGANIZATION'으로 변경	w0 f0 [0 0]

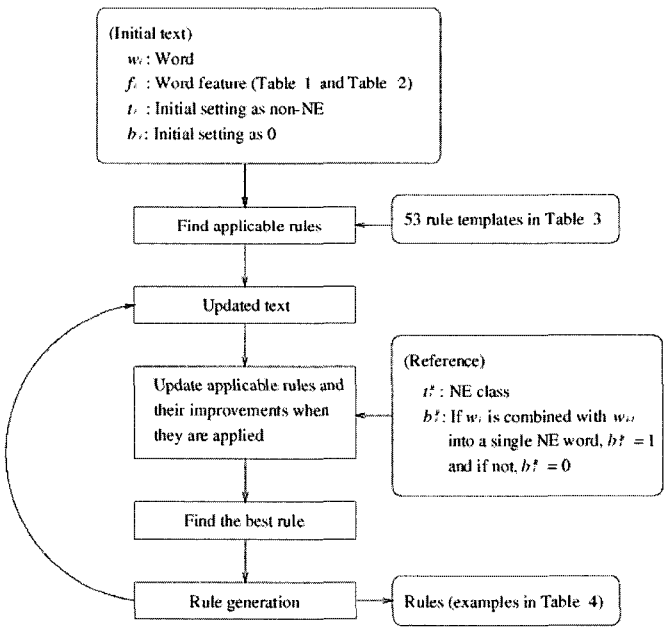
테스트시, 규칙들은 생성된 순서에 따라 입력 텍스트에 하나씩 적용이 된다. 규칙에 대한 조건이 맞는 경우, 해당 규칙이 적용되고, 이에 해당 단어들의 NE 타입이 변경된다.

테스트 자료에는 나오지만 학습자료에 없는 단어들에 대해 많은 주의가 필요하다. 이러한 상황에 대한 한 가지 대처 방법은 단어 리스트 미등록어들에 대해서 별도의 규칙들을 만드는 것이다. 학습자료의 데이터는 크게 두개의 그룹으로 나뉘어진다. 만약 단어들이 다른 그룹에서 나타나지 않는다면 해당 단어들을 단어 리스트 미등록어로 처리하고, 같은 규칙 생성 방식을 적용한다.

<그림 5>는 변환 기반 NE 규칙 자동생성에서의 전체 단계들을 보여주고 있다. 단계들은 초기값을 가진 텍스트로부터 시작된다. 이러한 초기값의 지정에 대한 구

체적인 방법은 3.1.1.에서 기술되었다. NE 타입과 NE 경계가 정확하지 않은 모든 단어들에 대해서, 이러한 NE 클래스와 NE 경계를 정확하게 해주는 규칙들이 생성된다. 규칙들은 <표 3>에 나타난 53개의 규칙 템플레이트에 의해서 생성된다.

모든 생성 가능한 규칙들 중에서, NE 타입과 NE 경계에 대해 현 텍스트와 답안 텍스트간의 에러들을 최대로 줄여주는 규칙이 입력 텍스트에 적용이 되고, 이에 따라 입력 텍스트가 업데이트 된다. 규칙 생성의 세부사항들은 3.1.2.에 기술되어 있다. 규칙들은 하나씩 생성이 되며, 이러한 단계들은 에러를 줄일 수 있는 규칙을 찾을 수 없을 때까지 계속 반복 된다. 생성된 규칙들의 예들은 <표 4>에 예시되어 있다.



<그림 5> 변환 기반 NE 규칙 자동 생성

4. 실험 및 결과

규칙 기반의 시스템의 성능을 측정하기 위해서, 규칙 기반의 시스템은 IdentiFinder와 비교되어 졌다. IdentiFinder는 BBN의 HMM 기반 시스템으로서, 1998 Hub-4 Broadcast News (BN) 벤치마크 테스트 [4][26]에 참여한 5개의 시스템 중 가장 좋은 성능을 보여준 시스템이다. 벤치마크 테스트에서의 결과와 비교할 때, 본 논문에서 얻은 IdentiFinder의 테스트 결과는 약간 다른데, 그 이유는 학습에 사용된 자료의 양이 다르고, 텍스트에 대한 전처리 과정들이 다르기 때문이다. 또한,

사용된 IdentiFinder의 버전이 다를 수도 있다.

두 시스템의 결과는 제일 처음 baseline 조건 (구두점 정보 없음, 대소문자 정보 없음, 네임 리스트 사용하지 않는 조건)에 대해서 비교된다. 그 이후, baseline에서 구두점과 대소문자와 같은 추가 텍스트 정보들로 인한 두 시스템의 성능향상 정도가 조사된다. 추가로, 네임 리스트의 효과도 두 시스템 모두에 대해서 논의되어진다. 마지막으로, 음성인식 에러에 따른 성능 저하와 두 시스템의 성능 저하 정도가 비교되어진다.

4.1. Baseline 조건에서의 실험 결과

100시간 분량의 1998 Hub-4 BN 데이터 세트가 규칙 기반 시스템과 IdentiFinder의 개발에 사용되어졌다. 평가는 NIST Hub-4 IE scoring pipeline package를 이용해서 3 시간 분량의 NIST 1998 Hub-4 BN 벤치마크 테스트 자료를 이용해서 수행되었다.

두 시스템들을 F-measure[27]와 Slot Error Rate(SER)[27]의 척도를 이용해서 평가하게 되는데, F-measure를 구하기 위해서 필요한 Precision(P)과 Recall(R), 그리고 F-measure(F)와 SER의 정의는 다음과 같다.

$$P = \frac{\text{정확한 } NE \text{ 개수}}{\textit{hypothesis}\text{의 } NE \text{ 개수}} \tag{1}$$

$$R = \frac{\text{정확한 } NE \text{ 개수}}{\textit{reference}\text{의 } NE \text{ 개수}} \tag{2}$$

$$F = \frac{RP}{(R+P)/2} \tag{3}$$

$$SER = \frac{\text{부정확한 } NE \text{ 개수}}{\textit{reference}\text{의 } NE \text{ 개수}} \tag{4}$$

Reference에 있는 NE와 인식기에 의해 생성된 NE간 비교에서 NE의 클래스는 동일하지만, NE의 영역이 일치하지 않고 overlap 되는 경우 정확한 NE의 개수에 0.5만 반영이 된다. 영역이 일치하지만, NE 클래가 일치하지 않는 경우에도 정확한 NE의 개수로 0.5만 반영된다.

규칙 기반 시스템의 성능은 Identifinder의 성능과 baseline 조건(구두점 정보 없음, 대소문자 정보 없음, name list 사용하지 않음)에서 비교되었다. 이 비교에서, 학습 및 테스트 텍스트는 대소문자 구분이 없고, 구두점이 없는 텍스트로 변환후

사용되었다. 그 후, 두 시스템 모두 네임 리스트를 사용하지 않고 학습되었다. <표 5>는 baseline 조건에서 각각의 시스템의 성능을 보여주고 있다. IdentiFinder와 비교했을 때, 규칙 기반 시스템은 F-measure로는 0.0012의 작은 성능 우위를 보였으나, SER로는 0.35%의 작은 성능 저하를 보여 주었다.

<표 5> Baseline 조건시 답안 텍스트에 대한 성능 비교 (RBS: 규칙 기반 시스템; IDF: IdentiFinder; SER: Slot Error Rate; Baseline: 구두점 없고, 대소문자 구분 없고, 네임 리스트 사용이 없는 조건)

조건	F-measure		SER (%)	
	RBS	IDF	RBS	IDF
Baseline	0.8858	0.8846	20.03	19.68

4.2. 구두점과 대소문자 정보의 효과

다음으로 구두점의 효과가 측정되었다. 두 시스템 모두 구두점을 별개의 단어로 취급해서 이용했다. 구두점 정보의 추가로 인한 성능 향상이 어느 정도인지를 측정하기 위해서, 두 시스템들은 구두점 정보가 포함된 텍스트로부터 학습이 되었다. 구두점은 NE 인식에 있어서 긍정적인 효과를 가지고 왔고, 규칙 기반 시스템의 성능을 F-measure로 0.0043, IdentiFinder의 성능을 0.0074만큼 높였다. SER로는 이러한 긍정적인 효과가 규칙 기반 시스템과 IdentiFinder에 대해 각각 0.93%, 1.29%로 측정되었다.

대소문자 정보의 효과 또한 측정되어졌다. 대소문자 정보의 포함이 두 시스템 모두에 어느 정도 성능 향상을 가져왔는지를 측정하기 위해서, 두 시스템 모두 대소문자 정보는 있지만, 구두점 정보가 없는 텍스트에서 학습되었다. 대소문자 정보는 NE 인식에 있어서 유용한 정보로 밝혀졌다. F-measure의 관점에서, 구두점 정보는 규칙 기반 시스템의 성능을 0.0146 향상 시켰고, IdentiFinder는 0.0154 향상 시켰다. SER 관점에서 구두점 정보는 규칙 기반 시스템의 성능을 3.48%, IdentiFinder의 성능은 3.08% 향상 시켰다.

<표 6>은 이 결과를 보여준다. 실험 결과에 따르면 대소문자 정보의 추가는 구두점 정보보다 시스템의 성능을 더 많이 올려준다.

<표 6> 구두점과 대소문자 정보의 효과 (SER: Slot Error Rate; RBS: 규칙 기반 시스템; IDF: IdentiFinder; Baseline+Capitalisation: 대소문자 정보는 있지만, 구두점 없고, 네임리스트 사용하지 않은 조건; Baseline+Punctuation: 구두점은 있지만, 네임 리스트 사용하지 않고, 대소문자 정보 없는 조건)

조건	F-measure		SER (%)	
	RBS	IDF	RBS	IDF
Baseline+Capitalisation	0.9004	0.9000	16.55	16.60
Baseline+Punctuation	0.8901	0.8920	19.10	18.39
Baseline	0.8858	0.8846	20.03	19.68

4.3. 네임 리스트의 효과

네임 리스트의 효과를 분석하기 위해서, 규칙 기반의 NE 인식기와 IdentiFinder가 대소문자 정보가 없고, 구두점이 없는 형태의 데이터에 대해서 학습이 되었다. 규칙 기반의 시스템과 같이, IdentiFinder는 네임 리스트로부터의 NE 정보를 hard-decision 하지 않고 단어 특성으로 활용한다[18]. 규칙 기반의 시스템이 네임 리스트로부터의 정보를 이용할 때, 만약 한 개 이상의 네임 리스트의 엔터티에서 중복이 일어나게 되면, 이 시스템은 긴 엔터티를 선호한다. 만약 동일한 단어가 한개 이상의 네임 리스트에서 발견되면, 네임 리스트간 우선순위가 적용된다. 장소명이 가장 높은 우선순위를 가지게 되고, 그 다음이 사람 이름, 그리고, 기관명이 가장 낮은 우선순위를 가지게 된다.

네임 리스트의 효과는 네임 리스트는 사용했지만, 구두점과 대소문자 구분은 사용하지 않은 경우에 측정되었다. F-measure 상으로, 네임 리스트의 사용은 규칙 기반 시스템의 성능을 0.0104 향상시켰고, IdentiFinder를 0.0108 향상시켰다. SER상으로 네임 리스트는 규칙 기반 시스템을 2.27% , IdentiFinder를 1.98% 향상시켰다.

<표 7> 네임 리스트의 효과. 실험은 baseline에 네임 리스트가 첨가된 조건에서 수행됨 (NL: 네임 리스트; RBS: 규칙 기반 시스템; IDF: IdentiFinder; SER: Slot Error Rate)

조건	F-measure		SER (%)	
	RBS	IDF	RBS	IDF
Baseline+NL	0.8962	0.8952	17.76	17.70
Baseline	0.8858	0.8846	20.03	19.68

<표 8>은 대소문자 정보와 구두점, 그리고 네임 리스트의 성능에 대한 효과를 정리해 주고 있다. 대소문자가 구분되고, 구두점이 있는 데이터가 네 가지의 다른 데이터로 가공되었다. 하나는 대소문자 구분, 구두점 있음, 다른 하나는 대소문자

구분, 구두점 없음, 또 다른 하나는 대소문자 구분 없음, 구두점 있음, 마지막 하나는 대소문자 구분 없음, 구두점 없음이다. 각각의 데이터에 대해서 규칙 기반의 시스템과 IdentiFinder는 네임 리스트가 있는 경우와 네임 리스트가 없는 경우 각각 학습이 되었다. 이러한 8가지의 각기 다른 학습 및 테스트 조건들은 <표 8>에 나타나 있다.

<표 8> 실험 결과 비교 (Cap: 대소문자 정보; NL: 네임 리스트; Punc: 구두점 정보; RBS: 규칙 기반 시스템; IDF: IdentiFinder; SER: Slot Error Rate)

조건	F-measure		SER (%)	
	RBS	IDF	RBS	IDF
Baseline+Cap+NL+Punc	0.9134	0.9145	13.98	14.15
Baseline+Cap+NL	0.9105	0.9121	14.72	14.30
Baseline+Cap+Punc	0.9086	0.9087	15.04	15.11
Baseline+Cap	0.9004	0.9000	16.55	16.60
Baseline+NL+Punc	0.9007	0.9010	16.68	16.69
Baseline+NL	0.8962	0.8952	17.76	17.70
Baseline+Punc	0.8901	0.8920	19.10	18.39
Baseline	0.8858	0.8846	20.03	19.68

추가정보(구두점과 대소문자 정보)를 사용하고, 네임 리스트를 함께 사용하게 되면, F-measure로 규칙기반 시스템이 0.0276, IdentiFinder는 0.0299가 향상된다. 구두점과 대소문자 정보, 그리고 네임 리스트로부터 NE 인식 시스템의 인식성능 향상 정도는 규칙 기반 시스템에 대해 F-measure로 각각 0.0043, 0.0146, 0.0104, IdentiFinder에 대해 F-measure로 각각 0.0074, 0.0154 and 0.0106으로 측정된다.

이 세 가지 추가 정보를 동시에 사용한 경우 두 시스템의 성능 향상 정도는 각각의 성능향상 정도의 합보다 조금 작게 측정되었다. 이 사실은 몇몇 NE 단어들은 네임 리스트와 함께 추가 제공된 구두점 및 대소문자 정보에 의해 정확히 인식될 수 있음을 보여준다.

놀랍게도, "Baseline+Cap+Punc"의 경우, baseline 조건으로부터 두 시스템의 실질 성능 향상 정도는 구두점 정보만 사용했을 경우의 성능향상 정도와 대소문자 정보를 사용했을 경우의 성능향상 정도의 합보다 크게 나왔다. 몇몇 NE 단어들은 NE 인식기에 의해서 정확히 인식되기 위해서 대소문자 정보와 구두점 정보가 동시에 제공되어야 하는 것으로 믿어진다. SER의 관점에서 결과를 분석해 보았을 때도 같은 결론을 얻을 수 있다.

NE 인식에서, SER은 (1.0 - F-measure)에 비례하게 된다. SER은 일반적으로 (1.0 - F-measure)보다 60%에서 70%정도 높게 나온다. <표 8>에서, 규칙기반 시스템은 F-measure상으로 IdentiFinder에 비해 조금 더 좋은 결과를 보였지만, SER에 대해서는 Baseline과 "Baseline+NL"의 경우에 대해서 조금 좋지 못한 결과를 보였

다. "Baseline+Cap+NL+Punc", "Baseline+Cap+Punc", "Baseline+NL+Punc"의 경우, 반대의 결과가 관찰된다.

<표 8>의 결과로부터, 두 시스템의 성능은 매우 비슷하며, baseline에서 각 조건에 대한 인식율 향상 정도도 거의 유사하다고 할 수 있다. 이 결과로부터, 두 시스템은 NE 인식에 대해서 비슷한 수준의 인식 능력을 가지고 있다고 결론 내릴 수 있다.

4.4. 음성인식 오류에 의한 Named Entity 인식율 변화

NE 인식을 위해서 학습된 패턴들은 다양한 문법적 그리고 의미적 구조를 반영하도록 디자인되어 있다. 따라서, 요구되는 패턴의 각 엘리먼트들은 입력 문장의 오류에 매우 민감하게 반응할 수 있다. 만일 요구되어지는 패턴의 엘리먼트가 없거나, 또는 만일 엘리먼트들 간에 필요 없는 엘리먼트가 첨가된 경우, 패턴은 입력 문장에 맞지 않게 된다. 음성인식 오류의 영향을 분석하기 위해서 1998 Hub-4 evaluation에 참여한 11개의 각기 다른 음성인식기 시스템의 결과를 이용하여 음성인식 오류에 대한 실험이 수행되었다. 이들 음성인식기의 결과는 NIST 웹사이트에서 얻을 수 있다[26].

규칙 기반 시스템과 IdentiFinder의 성능은 위의 11개 음성인식 시스템의 결과에 대해서 분석되어졌다. 실험은 구두점 정보와 대소문자 정보는 없지만 네임 리스트는 사용되는 경우에 대해서 수행되었다. 규칙 기반 시스템과 IdentiFinder는 사람이 수동으로 만든 학습 자료를 이용해서 학습시켰다. 결과들은 <표 9>에 나타나있으며, F-measure 결과 값들은 <그림 6>에 나타나 있다.

<표 9> 음성인식 오류에 따른 NE 인식율의 저하 정도 (WER: Word Error Rate; SER: Slot Error Rate; RBS: 규칙 기반 시스템; IDF: IdentiFinder)

System	WER (%)	F-measure		SER (%)	
		RBS	IDF	RBS	IDF
human transcription	0.0	0.8962	0.8952	17.76	17.70
ibm1	13.5	0.8051	0.8018	31.48	31.71
ibm2	13.6	0.8056	0.8003	31.28	32.27
limsi1	13.6	0.8146	0.8088	29.43	30.59
cu-htk1	13.8	0.8169	0.8099	30.46	31.05
ibm3	14.1	0.8012	0.7935	33.34	35.50
dragon1	14.5	0.8053	0.8059	31.33	32.03
bbn1	14.7	0.8096	0.7999	31.30	33.33
philips rwth1	17.6	0.7888	0.7878	34.92	34.69
sprach1	20.8	0.7618	0.7611	41.23	40.30
sri1	21.1	0.7700	0.7649	38.66	39.43

참 고 문 헌

[1] United States Defense Advanced Research Projects Agency (DARPA), Information Technology Office, "Named entity task definition", *Proc. 6th Message Understanding Conference*, pp. 317-332, 1995.

[2] N. Chinchor, "Overview of MUC-7/MET-2", *Proc. 7th Message Understanding Conference*, 1997. Available at http://www.muc.saic.com/proceedings/ muc_7_toc.html.

[3] B. Sundheim, "Overview of results of the MUC-6 evaluation", *Proc. 6th Message Understanding Conference*, pp. 13-31, 1995.

[4] M. Przybocki, J. Fiscus et al., "1998 Hub-4 information extraction evaluation", *Proc. DARPA Broadcast News Workshop*, pp. 13-18, 1999.

[5] W. Black, F. Rinaldi, D. Mowatt, "FACILE: description of the NE system used for MUC-7", *Proc. 7th Message Understanding Conference*, 1997. Available at http://www.muc.saic.com/proceedings/muc_7_toc.html.

[6] H. Chen, Y. Ding et al., "Description of the NTU system used for MET2", *Proc. 7th Message Understanding Conference*, 1997. Available at http://www.muc.saic.com/proceedings/muc_7_toc.html.

[7] J. Fukumoto, F. Masui et al., "Oki electric industry: description of the Oki system as used for MUC-7", *Proc. 7th Message Understanding Conference*, 1997. Available at http://www.muc.saic.com/proceedings/muc_7_toc.html.

[8] R. Yangarber, R. Grishman, "NYU: description of the Proteus/PET system as used for MUC-7", *Proc. 7th Message Understanding Conference*, 1997. Available at http://www.muc.saic.com/proceedings/muc_7_toc.html

[9] D. Miller, R. Schwartz et al., "Named entity extraction from broadcast news", *Proc. DARPA Broadcast News Workshop*, pp. 37-40, 1999.

[10] D. Palmer, J. Burger, M. Ostendorf, "Information extraction from broadcast news speech data", *Proc. DARPA Broadcast News Workshop*, pp.41-46, 1999.

[11] S. Renals, Y. Gotoh et al., "Baseline IE-NE experiments using the SPRACH/LASIE system", *Proc. DARPA Broadcast News Workshop*, pp. 47-50, 1999.

[12] D. Appelt, D. Martin, "Named entity extraction from speech: approach and results using the TextPro system", *Proc. DARPA Broadcast News Workshop*, pp. 51-54, 1999.

[13] J. Kim, P. Woodland, "Rule based named entity recognition", *Technical Report CUED/F-INFENG/TR.385*, Cambridge University Engineering Department, 2000.

[14] S. Katz, "Estimation of probabilities from sparse data for the language model component of a speech recognizer", *IEEE Transactions on Acoustics, Speech and Signal Processing*, Vol. 35, No. 3, pp. 400-401, 1987.

[15] L. Rabiner, B. Juang, "An introduction to hidden Markov model", I*EEE Acoustics, Speech and Signal Processing Magazine*, Vol. 3, pp. 4-16, 1986.

[16] L. Rabiner, B. Juang, *Fundamentals of Speech Recognition*, Prentice Hall, 1993.

[17] S. Young, "Large vocabulary continuous speech recognition: a review", *IEEE Signal Processing Magazine*, 1996.

[18] D. Bikel, S. Miller, R. Schwartz, "Nymble: a high-performance learning name-finder", *Proc.*