논문-98-3-2-09

문법적 제약을 이용한 금액 문장 인식의 성능 향상

함 정 표*, 양 태 영*, 신 원 호*, 이 충 용*, 차 일 환*

Improvement of Price Sentence Recognition Using Grammatical Constraint

Jeong-Pyo Ham*, Tae-Young Yang*, Won-Ho Shin*, Chungyong Lee*, and Il-Whan Cha*

요 약

연속음 인식에서 인식 대상이 가지는 규칙을 적용했을 경우 성능 향상을 가져올 수 있다. 본 논문에서는 연속음 중에서 연결 숫자음을 인식 대상으로 하는 음성 인식 시스템의 성능 향상을 위하여 프레임 동기 네트워크(Frame Synchronous Network)을 이용하였다. 연결 숫자음이 가지는 반복적인 특성과 자릿수의 상하 관계가 인식 성능에 미치는 효과를 이용하여 다양한 수준의 제약을 갖는 FSN을 제안하였다. 본 논문에서는 연속 숫자음 중에서 금액을 대상으로 인식 결과 제안된 FSN을 이용하여 금액 어휘의 인식 성능을 향상시킬 수 있었다.

Abstract

Using the grammar of sentences can alleviate performance of connected word recognition. In this paper, in order to improve the recognition rate of the connected digit recognition system Frame Synchronous Network is used. The proposed FSN's are based on the various grammatical characteristics of connected digits such as repetition and position. Experiments were performed on Korea price sentences that are composed of connected digits and the results showed that the recognition rate is improved.

I. 서 론

연결음은 몇 개의 단독음이 연결된 형태로 모델링할 수 있다. 이러한 연결음(connected word)의 경우는 인식 대상 연결음 문장이 몇 개로 이루어져 있으며, 구성하는 단독음 단어들이 어디에서 경계를 이루며 어떤 단어들로 이루어져 있는가 하는 문제를 해결해야 한다. 이러한 문제들을 해결하기 위해서 두 단계(two level) 알고리즘[1], 단계 쌓기(level building) 알고리즘[2], 단일 경로(one path) 알고리즘[3] 등이 있다. 이 중에서 단일 경로 알고리즘은 입력 프레임에 동기적(synchronous)으로 구현할 수 있을 뿐만 아니라, 적은 계산량으로 처리할 수 있어서 매우 우수한 알고리즘이라고 할 수 있다.

본 논문에서 사용된 연속 숫자음 중의 하나인 금액은 특정한 규칙을 가지고 발음되고 이것을 이용하여 인식률을 향상 시킬 수 있다. 이러한 문법적 특징을 구조적으로 시험 문장을 모델링하는 방법으로 FSN이 있다. 본 논문에서는 금액 인식을 위하여 연결음 인식 대상 중에 하나인 금액의 문법적 특성을 분석하고, 분석된 규칙들에 적합한 형태의 FSN을 제안하여 인식률 향상을 목표로 하였다.

2장에서 FSN 알고리즘에 대하여 살펴보고, 3장에서 본 논문의 인식 대상인 금액의 문법적 특징에 대하여 적는다. 4장에서 3장에 적은 규칙을 바탕으로 FSN을 제안하며, 5장에서 제안된 FSN으로 실험 결과를 적고 6장에서 결론을 맺는다.

* 연세대학교 전자공학과
Dept. of Electronic Engineering Younsei University
※ 본 논문은 한국통신연구개발본부의 1998년도 수탁과제연구지원에 의한 결과입니다.

II. Frame Synchronous Network Algorithm

단일 경로 알고리즘에서 단어간 천이를 제어함으로써 연속음 인식에 문법을 도입할 수 있다. 연이은 두 단어가 천이할 때 확률적으로 가중치를 두는 bi-gram이 대표적 예이다. FSN(frame synchronous network)은 문법을 문장이 발생할 수 있는 구조를 이용하여 구현한다[4]

FSN은 절점(node)과 호(arc)로 이루어진다. 네트워크의 절점은 단어의 경계가 되고 호는 단어 모델들로 이루어진다. 단일 경로 알고리즘에서 단어의 중간 상태에서 이루어지는 단어 내부의 천이는 FSN의 호에서 수행되고, 단어의 첫 상태에서 이루어지는 단어간의 천이는 절점을 중심으로 수행된다. 간단히 FSN을 한 단계 높은 HMM이라고 간주할 수 있고, FSN의 절점은 HMM의 상태(state)로 대응되고, FSN의 호는 HMM의 상태간 천이(state transition)로 대응된다. HMM의 상태가 확률 분포를 포함하고 있는 반면에 FSN은 호에서 확률 분포를 가지는 차이점이 있다. HMM과 같이 확률 분포 함수에 의하여 관찰 확률을 얻는 것이 아니라, 관찰열이 호를 통과하며 호 내부에서 단어 수준의 비터비 디코딩을 사용하여 각 호의 관찰 확률이 얻어진다. 단일 경로 알고리즘과 같이 단어의 FSN의 경로도 DP를 이용하여 관찰열로부터 알아낼 수 있다.

다음 그림에 간단한 FSN의 한 예가 있다.

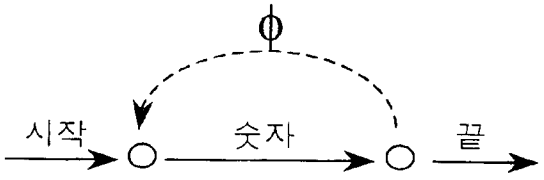


그림 1. 간단한 FSN의 예
Fig. 1. An example of a simple FSN

위의 네트워크는 임의 개수의 연속 숫자 문장을 인식할 수 있는 네트워크이다. 상위 수준의 HMM인 FSN 또한 비터비 디코딩에 의하여 확률적으로 최적인 경로를 찾아 어떠한 단어로 구성된 문장인지 인식하게 된다. FSN에서의 디코딩은 역추적을 통하여 경로를 찾은 후, 어떤 단어가 입력되었는지 결정한다는 관점에서 경로 조합(Path Merge)이라고 할 수 있다. 본질적으로는 비터비 알고리즘과 동일한 DP의 확장이다. 단지 일반적으로 많이 사용되는 HMM과 달리 네트워크에서는 무효인 호(null arc)가 존재한다는 것이 차이점이다. 무효 호를 사용하면 확률적인 손실 없이 다른 절점으로 연결할 수 있다. 위에 예를 든 네트워크의 경우, 2개의 절점이 무효 호로 연결되어 한 개부터 무한 개까지 단어로 이루어진 문장 후보 중에 최대 우도(maximum likelihood) 방법에 의하여 가장 발생할 확률이 큰 문장으로 인식되는 것이다.

FSN의 전체적인 탐색 알고리즘을 보면 다음과 같다.

전체적인 FSN 탐색 알고리즘

1) 모든 절점과 경로 추적 버퍼를 초기화
2) 모든 프레임에 대하여 최적 경로 탐색을 수행
   특징 벡터 추출
   모든 단어 모델의 천이 가능한 상태에 대하여 관찰 확률 계산
   for 입력 신호의 모든 프레임에 대하여
   for 네트워크의 모든 천이 가능한 절점에 대하여
   for 절점 앞에 오는 모든 문법 절점에 대하여
   모든 내부 상태에 대하여 DP수행
   누적 확률과 경로 정보 갱신 (단어 수준)
   end
   경로 조합 DP 수행
   누적 확률과 경로 정보 갱신 (문장 수준)
   누적 경로 추적 버퍼 갱신
   end
   end
3) 후처리와 경로 추적을 수행
4) 인식 결과 결정

단어 수준의 DP는 앞서 논의한 비터비 알고리즘에 의하여 수행되는 HMM의 디코딩이고, 문장 수준의 DP는 절점과 호를 가지고 경로 조합(path merge)에 의하여 수행된다.

경로 조합 과정은 비터비 디코딩과 같이 최대 우도(maximum likelihood) 방법에 의하여 가장 발생 확률이 높은 경로를 찾는다. 경로 조합 과정 중 한 절점에서의 동작을 아래 그림에서 볼 수 있다.

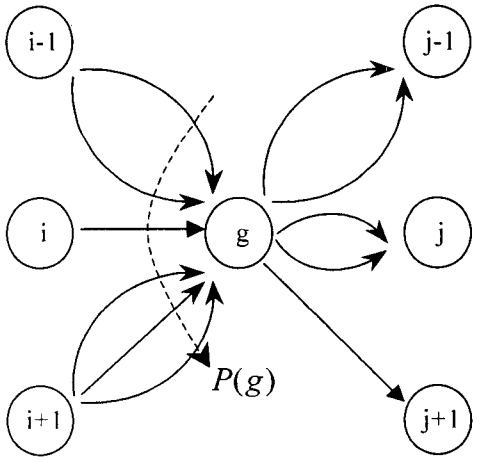


그림 2. 절점 g에서의 동작
Fin. 2. Path merge at node g

이전 절점에서 출발한 절점 g에 들어오는 모든 호에 대

하여 출력 확률을 구한다. 각 호의 내부는 앞서 비터비 알고리즘으로 확률을 계산하여 출력하고, 이들 확률값 중에서 가장 큰 값을 선택하여 시간 t에서의 $P(g)$로 할당한다. 동시에 최고 확률을 출력했던 호와 그 호가 시작한 절점은 나중에 있을 경로 추적을 위해 임시 변수에 저장한다. 이 과정을 정리하면 아래의 경로 조합(path merge) 과정과 같다.

Grammar Node g에서의 경로 조합 알고리즘

1) 시간 t에 문법 절점 g에 도달하는 모든 경로에 대하여 경로 조합 수행

$$glike(g,t)=\max_{k\le P(g)}[outbest(g,k)]$$
$$inbest(g)=glike(g,t)$$

2) 시간 t에 g에 도달하는 최적 경로의 역추적 정보 저장

$$word(g,t)=\arg\max_{k\le P(g)}[outbest(g,k)]$$
$$bp(g,t)=elapse[word(g,t)]$$
$$bpnode(g,t)=LG[word(g,t)]$$

여기서,

$word(g,t)$: 시간 t에 절점 g로 도달하는 누적 우도

$bp(g,t)$: 시간 t에 절점 g로 도달하는 마지막 호에 해당하는 단어

$bp(g,t)$: 시간 t에 절점 g로 도달하는 마지막 호의 지속 시간

$bpnode(g,t)$: 시간 t에 절점 g로 도달하는 마지막 호가 출발한 절점

$outbest(g,k)$: k번째 호를 통해 절점 g로 도달하는 최적 경로의 우도

$inbest(g)$: 절점 g들어가는 최적 우도

$elapse(i)$: 절점 i에 들어간 후 최적 경로를 통한 지속 시간

경로 조합에 의하여 각 시간 t에서 각 절점에 입력되는 경로 정보가 저장되었다. 네트워크의 끝 절점에서부터 최고 확률을 가지는 절점을 추적하면 모든 문장 후보 중에서 최고 확률을 가지는 문장이 결정되고, 이러한 과정을 경로 역 추적 과정(trace back)이라고 한다. 아래에서 경로 역 추적 알고리즘을 볼 수 있다.

경로 역 추적 알고리즘

1) 초기화 $i=g,\ j=t,\ k=0$

2) $j\neq 0$인 동안

$k=k+1$

$iword(k)=word(i,j)$

$idur(k)=bp(i,j)$

$i=bpnode(i,j)$

$j=j-idur(k)$

여기서,

$iword(k)$: 끝에서 k번째 인식되는 단어

$idur(k)$: 끝에서 k번째 인식되는 단어의 지속 시간

III. 금액의 문법적 특징 분석

이 절에서는 연결 숫자음의 하나인 금액의 문법적 특성을 살펴본다. 모든 금액 문장은 단위와 숫자로 구성되고, 이들은 번갈아 나타난다. 금액을 구성하는 단위와 숫자는 각각 독특한 문법적인 특성을 지니고 있다.

먼저 단위에 대하여 살펴보면, 단위로 사용되는 단어는 다음의 것들이 있다. {/경/, /조/, /억/, /만/, /천/, /백/, /십/, /원/ }. 하지만, 실험에 사용된 시스템의 최고 인식 단위는 수 천억으로, 앞의 집합에서 /경/, /조/는 인식 대상에 포함되지 않는다고 가정한다. 일반적으로 구현되는 시스템의 경우 이 가정이 충분히 합당하다.

단위는 크게 두 가지로 분류된다. 첫째는 4자리 숫자마다 반복되어 만 자리씩 끊어주는 /억/, /만/ 등의 '만단위'이다. 이 '만단위' 사이에 오는 구조인 '수천 수백 수십 수'는 모든 만 자리마다 동일하게 반복된다. 둘째는 '만단위' 사이에서 자릿수를 표기하는 /천/, /백/, /십/의 '자리 단위'가 있다. '만단위' 끼리와 '자리 단위' 끼리는 서로 순서가 바뀌어서 나올 수 없다. 즉, /억/ 〉 /만/ 이고, /천/ 〉 /백/ 〉 /십/ 으로 서로 크기 관계가 있으며, 이 순서가 바뀌어 나오는 문장은 문법적으로 잘못된 문장이 된다. 그러나, '만단위'와 '자리단위' 사이에서는 서로 섞여 나오는 것이 문법적으로 잘못된 문장은 아니다. 이것은 문법적인 오류를 말하는 것이지 인식된 결과가 정확하다는 것을 뜻하진 않는다.

다음에 그 예가 있다.

/사백 오천원/(틀림), /오천 사백원/(맞음) : /백/과/천/의 순서오류

/이십 팔만원/(맞음), /팔만 이십원/(맞음) : /십/과/만/의 순서 바뀜 허용

또한 특수한 단위로 /원/이 있다. /원/은 모든 금액 문장의 끝에 붙어 나오는 것으로 금액임을 표시한다. /원/의 경우 '만단위'의 하나로 가장 가치가 작은 것으로 해석하여 일반화 할 수 있다.

숫자는 다음의 아홉 가지 발음을 인식 대상으로 한다. { /일/, /이/, /삼/, /사/, /오/, /육/, /칠/, /팔/, /구/ }. 숫자를 표기할 때 '0' (= /영/ 혹은 /공/)은 표기되지만 일반적으로 금액에서 발음되지 않고 강조하는 경우에만 발음된다. 본 논문에서 구현한 시스템에서는 발음되지 않는 것을 원칙으로 한다. 예를 들면, '₩ 30,200' 은 /삼만 이백원/이라 발음하고 /삼만 영천 이백원/ 이라고 발음하지 않는다.

특수한 숫자로 '1' (= /일/)이 있다. /일/의 경우 '자리단위' 앞에서는 발음되지 않으며 4자리마다 반복되는 '만단위' 앞에서만 발음된다. 하지만, 가장 앞에 오는 /일/

은 '만단위' 앞이라 하더라도 일반적으로 발음되지 않는다. 혹은 발음하는 경우도 있지만 /만/의 경우 주로 발음되지 않고 /억/의 경우는 발음된다.

다음에 그 예가 있다.

'₩ 310,000'= /삼십 일만원/(맞음), /삼십 만원/ (틀림)

'₩ 15,000' = /만 오천원/(맞음), /일만 오천원/ (맞지만 드물다)

'₩ 150,000,000' = /일억 오천 만원/(맞음), /억 오천 만원/ (틀림)

IV. 제안된 FSN

앞서 고찰한 금액의 문법 구조를 이용하여 다음의 세 가지 유형의 FSN을 제안한다. 이들은 각기 다른 정도의 문법적 제한 사항을 두는 점에 차이가 있다. 문법적인 제약이 작은 순서대로 FSN0, FSN1, FSN2라고 한다.

FSN의 호를 이루는 인식 대상 단어들을 각각 하나의 이산 HMM으로 구성하였다[6]. 이들은 숫자 집합과 단어 집합의 합집합이다. 그리고, 단어 사이에 포함되는 묵음을 적절히 표현하기 위하여 묵음 모델을 하나 추가 하였다. {/억/, /만/, /천/, /백/, /십/, /원/, /일/, /이/, /삼/, /사/, /오/, /육/, /칠/, /팔/, /구/, '묵음'} 앞에 수록된 단어 모델들은 네트워크의 절점 사이를 잇는 호로 사용되며 대부분 숫자는 병렬로 사용된다.

1. FSN0

제안된 FSN 중에서 가장 문법적인 제한을 두고 있지 않은 유형으로 FSN0를 아래 그림 3에서 볼 수 있다. FSN0는 무한 개수의 숫자와 단위가 무작위로 섞인 인식 결과가 나올 수 있다. 즉, 임의의 개수를 가지는 단위와 숫자의 조합을 인식할 수 있지만, 숫자나 단위만 연속적으로 인식된다든지 혹은 낮은 단위가 높은 단위보다 먼저 나오는 것과 같은 문법적인 오류가 있는 문장이 결과로 나올 수도 있다.

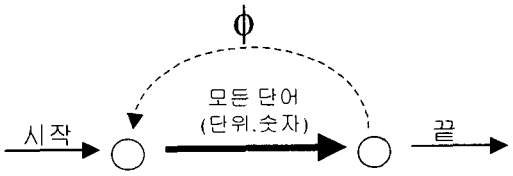


그림 3. 'FSN0'의 구조
Fig. 3. The structure of 'FSN0'

2. FSN1

FSN1을 제안하기 전에 먼저 sub-FSN을 정의 한다. 단어 모델 뿐만이 아니라 다른 FSN 또한 그보다 높은 단계 FSN의 호로 사용할 수 있다. 한 단계 아래에서 다른 FSN의 부분을 이룬다고 하여 sub-FSN이라고 부른다. 문법적으로 '만단위'와 '자리 단위' 중에서 '만단위' 앞에서만 나오는 특수한 숫자인 '1'을 FSN에서 구현할 때, 편의를 위해 다음의 두 가지 sub-FSN을 정의 한다. '1'을 포함하여 무효 호가 없는 '숫자1' FSN과, 1을 표현하지 않고 1 대신 무효 호를 포함하는 '숫자2' FSN을 정의한 그림이 아래 그림 4와 그림 5에 있다. '숫자1'은 단어 모델 '1', '2', '3', ... '9'가 병렬로 양쪽 노드에 연결되어 같은 위치에서 인식될 수 있는 숫자를 표현하고 있는 형태이다. '숫자2'는 '1'대신 무효 호가 병렬로 연결되어 '1'이 인식되는 경우는 '1'이 발음되지 않는 경우의 문법적인 특성을 고려한 형태이다. 앞서 금액의 문법적인 특성을 살펴본 바와 같이 '숫자1' 네트워크는 '만단위' 앞에 오고 '숫자2' 네트워크는 '자리단위' 앞에 온다.

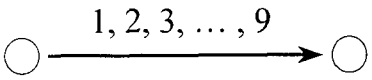


그림 4. Sub-FSN '숫자1'의 구조
Fig. 4. The structure of sub-FSN 'digit1'

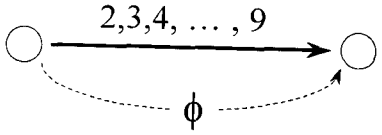


그림 5. Sub-FSN '숫자2'의 구조
Fig. 5. The structure of sub-FSN 'digit2'

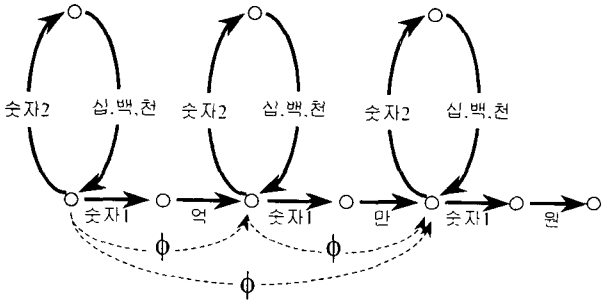


그림 6. 'FSN 1'의 구조
Fig. 6. The structure of 'FSN1'

그림 6는 '만단위'마다 반복되는 금액의 특성을 이용하여 제안된 'FSN1'이다. 이것은 FSN0에 비교하여 발생할 수 있는 문법적인 오류를 많이 제한한다. '만단위' 사이에는 루프를 두어 '숫자 + 자리단위'의 구조가 반복되는 것을 이용하였다. 따라서, 숫자나 단위만 반복해서 인식 결과가 될 수 있는 오류를 막았다. '만단위' 마다 연결된 무효 호는 최고 자리가 어느 만 단위에 속해 있는지 알 수

없기 때문에 연결되었다. 곧, 최고 자리가 '수 백만'이면, 최적 경로는 앞의 '억'으로 끝나는 만 단위는 무효 호를 통해 뛰어넘게 된다. 이와 같이 FSN은 연결 구조만으로 문법을 구현함으로써 인식률의 향상을 가져올 수 있다. FSN1 역시 많이 제한되었지만 '자리단위'의 상하 관계에서 문법적 오류가 발생할 수 있다. 예를 들어, '수백 수천 수십'으로 인식된 결과가 출력될 수 있다.

3. FSN2

FSN2를 정의하기 전에 subnet(x)를 정의한다. FSN1에서 만 단위 사이의 루프를 펼쳐서 하나의 네트워크로 구성한 것을 subnet(x)로 정의한다. 변수 x는 임의의 '만단위'가 대치하게 된다. 어느 자리에서 시작하는지 모르며, 중간에 '0'이 있는 자리는 단위까지 모두 발음이 되지 않으므로 모두 무효 호로 연결하여 다른 절점으로 중간 호를 거치지 않고 천이할 수 있도록 하여, '수천 수십'과 같이 중간에 '수백'이 '0'에 의해 생략된 경우를 표현할 수 있도록 하였다. subnet(x)는 '자리단위'가 일렬로 연결된 형태이므로, 상하 관계에 의한 인식 결과의 오류는 발생할 수 없다.

그림 7에 subnet(x)의 구조가 나타나 있다.

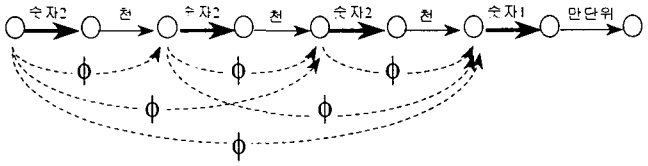


그림 7. subnet('만단위')의 구조
Fig. 7. The structure of subnet(×)

그림 8에는 subnet(x)을 이용하여 FSN2를 정의하였다. FSN2는 앞 절에 기록한 금액의 모든 문법적인 구조를 활용하여 문법적인 오류를 가지는 결과는 출력될 수 없으며, 오류가 없는 후보를 대상으로 인식하게 되므로 세가지 네트워크 중에서 가장 높은 인식률을 보이게 된다. '만단위'마다 규칙적인 subnet(x)이 반복적으로 나타나며, FSN1과 같이 자릿수 시작을 임의의 위치에서 가능하도록 하기 위하여 무효 호를 두었다.

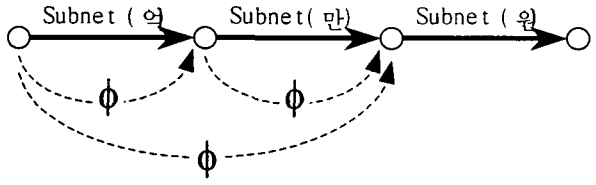


그림 8. 'FSN2'의 구조
Fig. 8. The structure of 'FSN2'

V. 실험 및 결과

실험에 사용되는 데이터 베이스를 구축하기 위하여 남녀 모두 50명을 대상으로, 각각 한 명의 화자가 40개의 금액 문장을 발음하였다. 이중에서 남녀 각 20명씩 모두 40명이 발음한 문장을 학습 데이터 베이스로 사용하고, 나머지 10명의 문장을 시험 데이터 베이스로 사용하였다. 금액은 천억 단위까지 인식이 가능하도록 수집되었으며, 발음되는 문장은 숫자가 모두 비슷한 횟수로 발음되게 하였으며, 모든 경우가 포함될 수 있도록 임의적으로 구성되었다. 아래 표 1에 데이터 베이스의 구성이 요약되어 있다.

표 1. 데이터 베이스의 구성 요약
Table 1. Database

인식어휘	한국어 금액, 최고 단위 수 천억
샘플링 방법	16bit linear PCM, 11.025 kHz sampling
학습 데이터	20대 남, 녀 각 20명(총 40명), 1인당 40문장
시험 데이터	20대 남, 년 각 5명(총 10명), 1인당 40문장

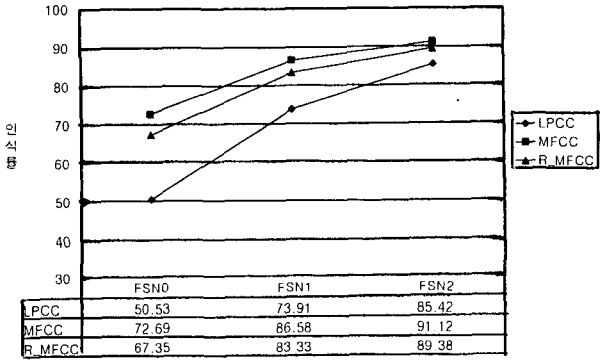
실험에 사용되는 음성 신호는 11.025 kHz로 표본화 되고 $1-0.95z^{-1}$의 전달 함수를 갖는 프리엠퍼시스 필터를 사용하여 고역을 강조한다. 신호의 분석은 매 10ms마다 20ms에 해당하는 220 샘플의 길이를 갖는 해밍 윈도우(hamming window)를 사용하여 수행되었다[7]. 멜 켑스트럼(MFCC)와 루트 멜 켑스트럼(R_MFCC)의 경우 각 음성 프레임마다 1024 point FFT를 사용하였다[5]. 켑스트럼이 구해지면 동적 특성을 이용하기 위하여 델타 켑스트럼을 구하여 사용하였다. 에너지에 대하여도 동일한 방법을 취하여 각각 델타 에너지와 델타 델타 에너지를 구하여 함께 사용하였다. 앞으로 켑스트럼 만을 사용하여 모두 12차의 특징 벡터를 사용한 경우를 cep이라고 표시하고, 켑스트럼과 델타 켑스트럼 합게 사용하여 모두 24차를 사용한 경우를 cep+d_cep이라고 표시하며, 켑스트럼, 델타 켑스트럼, 델타 에너지, 그리고 델타 델타 에너지를 함께 사용하여 모두 26차의 특징 벡터를 사용한 경우를 cep+d_cep+ eng라고 표기한다. 본 논문에서 사용된 음성 분석 방법 및 특징 벡터를 정리하면 다음 표 2와 같다.

표 2. 음성 분석 방법 및 특징 벡터
Table. 2. Speech analysis and Feature vector

Pre-Emphasis	1-0.85z-1
Window Size	220 samples(20ms)
Shift Size	110 samples(10ms)
Window	Hamming Window
FFT Size	1024
mel 뱅크 수	22
특징벡터	켑스트럼(12차)
	델타 켑스트럼(12차)
	델타 에너지+델타델타 에터지(2차)

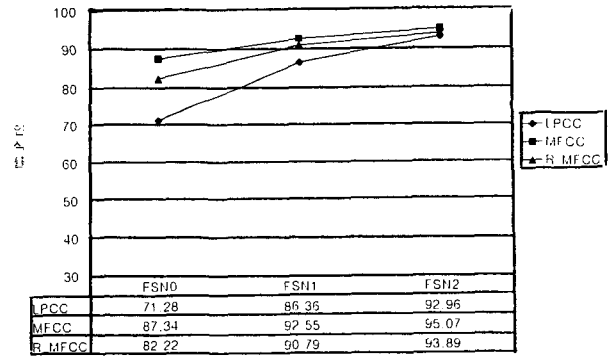
제안된 FSN을 이용하여 인식 대상인 금액의 규칙에 의한 성능을 평가하기 위하여 LPCC, MFCC, 그리고 R_MFCC의 세 가지 특징 벡터를 대상으로 FSN0, FSN1, FSN2를 사용하여 각각 실험하였다. 아래 그림에서 그림 9은 특징 벡터로 켑스트럼만 사용한 경우이다. 그림 10은 켑스트럼과 델타 켑스트럼을 특징 벡터로 사용하고, 그림 11는 켑스트럼, 델타 켑스트럼, 그리고 델타 에너지와 델타 델타 에너지 까지 모두 사용한 결과이다.

결과에서 볼 수 있듯이 문법적 제약이 많을수록 성능이 보다 향상되는 것을 알 수 있다. 특히 금액의 반복적인 규칙을 사용하는 경우 아무 것도 사용하지 않는 경우보다 성능이 많이 향상된다. 자릿수에 의하여 상하 관계가 생기는 규칙까지 사용한 경우는 성능이 더욱 향상되어 아무 문법도 사용하지 않은 경우와 최고 30%의 성능 개선을 관찰할 수 있다.



	FSN0	FSN1	FSN2
LPCC	50.53	73.91	85.42
MFCC	72.69	86.58	91.12
R_MFCC	67.35	83.33	89.38

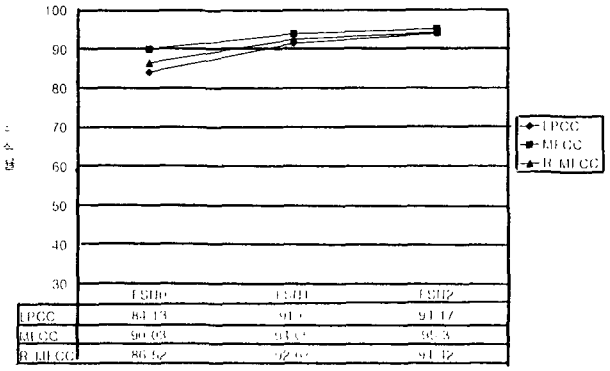
그림 9. FSN에 따른 성능 비교(cep)
Fig. 9. Performance improverment by FSN(cep)



	FSN0	FSN1	FSN2
LPCC	71.28	86.36	92.96
MFCC	87.34	92.55	95.07
R_MFCC	82.22	90.79	93.89

그림 10. FSN에 따른 성능 비교(cep+d_cep)
Fig. 10. Performance improvement by FSN(cep+d_cep)

또한 특징 벡터를 많이 사용할수록 성능이 좋아지는 것을 관찰할 수 있다. 특징 벡터를 많이 사용할수록 성능이 좋아지는 것은 문법이 약한 경우 두드러지게 나타난다. FSN2의 경우는 특징 벡터를 많이 사용하여 인식률이 향상되는 폭이 FSN0나 FSN1과 비교하여 작다.



	FSN0	FSN1	FSN2
LPCC	84.13	91.0	91.17
MFCC	90.03	91.0	95.3
R_MFCC	86.52	92.0	91.12

그림 11. FSN에 따른 성능 비교(cep+d_cep+eng)
Fig. 10. Performance improvement by FSN(cep+d_cep+eng

VI. 결 론

본 논문에서는 금액의 반복적인 규칙과 상하 관계의 규칙을 이용하여 인식 성능 향상을 연구하였다. 여러 가지 종류의 특징 벡터와 델타 켑스트럼과 델타 에너지, 델타 델타 에너지를 함께 사용한 경우를 대상으로 실험한 결과, 모든 경우에서 문법 제약은 인식 성능을 향상시키는 결과를 얻었다. 그 중에서도 FSN1을 통하여 문법적인 규칙을 일부만 사용하여서도 인식 성능 향상을 얻을 수 있었다. 최대 성능 향상은 LPCC를 특징 벡터로 켑스트럼만 사용한 경우 문법을 사용하지 않는 50.53%에서 모든 문법을 사용하는 85.42%로 약 30%이상 성능 향상이 있었다.

참 고 문 헌

[1] H. Sakoe, "Two level DP matching a dynamic programming-based pattern matching algorithm for connected word recognition," *IEEE Trans. on Acoust., Speech, and Signal Processing*, vol. ASSP-27, no. 6, Dec. 1979.

[2] C. S. Myers and L. R. Rabiner, "A level building dynamic time warping algorithm for connected word recognition," *IEEE Trans. on Acoust., Speech, and Signal Processing*, vol. ASSP-29, no. 2, Apr. 1981.

[3] H. Ney, "The use of a one-stage dynamic programming algorithm for connected word recognition," *IEEE Trans. on Acoust., Speech,, and Signal Processing*, vol. ASSP-32, no. 2, Apr. 1984.

[4] C. H. Lee and L. R. Rabiner, "A frame-

synchronous network search algorithm for connected word recognition," *IEEE Trans. on Acoust., Speech, and Signal Processing*, vol. 37, no. 11, Nov. 1989.
[5] L. R. Rabiner and B. H. Juang, *Fundamentals of Speech* Recognition, Prentice Hall, 1993.
[6] X. D. Huang, Y. Ariki and M. A. Jack, *Hidden Markov Models for Speech* Recognition, Edinburgh University Press, 1990.
[7] E. C. Ifeachor and B. W. Jervis, *Digital Signal Processing*, Addison-Wesley Publishing Company Inc., 1993.

저 자 소 개

함 정 표

1993년 3월 ~ 1997년 2월 연세대학교 전자공학과(공학사)
1997년 3월 ~ 현재 : 연세대학교 대학원 전자공학과 석사과정

양 태 영

1989년 3월 ~ 1993년 2월 연세대학교 전자공학과(공학사)
1993년 9월 ~ 1995년 8월 연세대학교 대학원 전자공학과(공학석사)
1995년 9월 ~ 현재 연세대학교 대학원 전자공학과 박사과정

신 원 호

1986년 3월 ~ 1991년 2월 연세대학교 전자공학과(공학사)
1992년 3월 ~ 1994년 2월 연세대학교 대학원 전자공학과(공학석사)
1994년 3월 ~ 현재 연세대학교 대학원 전자공학과 박사과정

이 충 용

1983년 3월 ~ 1987년 2월 연세대학교 전자공학과(학사)
1987년 3월 ~ 1989년 2월 연세대학교 전자공학과(석사)
1990년 3월 ~ 1991년 8월 연세대학교 산업기술 연구소 연구원
1991년 9월 ~ 1995년 12월 Ph. D., Georgia Institute of Technology, At lant, GA
1996년 2월 ~ 1997년 7월 삼성전자 선임연구원
1997년 9월 ~ 현재 연세대학교 기계전자공학부 조교수

차 일 환

1955년 3월 연세대학교 공과대학 공학사
1959년 2월 연세대학교 전기공학 공학석사
1969년 ~ 1970년 영국 Soulthampton University Institute of Sound and Vibration Research 수학 음향공학
1983년 2월 연세대학교 대학원 전자공학과 음향공학 공학박사