한국퍼지 및 지능 시스템학회 논문지 1998, Vol. 8, No. 2. (1998)82~95

지연누적에 기반한 화자결정회로망이 도입된 구문독립 화자인식시스템

Text-Independent Speaker Identification System Using Speaker Decision Network Based on Delayed Summing

이 종 은, 최 진 영
Jong-Eun Lee, Jin Young Choi
서울대학교 전기공학부, ERC-ACI, ASRI

요 약

본 논문에서는 구문독립 화자인식 시스템에서 가장 중요한 역할을 하는 분류기를 두 단계로 나누어, 먼저 짧은 구간들에 대해서 각각의 화자에 속하는 정도를 계산하고, 다음에 계산된 결과들을 가지고 주어진 음성구간전체에 대해 가장 가능성이 높은 화자를 선택하는 구조를 제안한다. 첫번째 부분은 학습에 의해 스스로 조직하는 RBFN을 이용하여 구현하고 두번째 부분에서는 MAXNET과 지연합의 조합으로 화자를 결정한다. 이렇게 함으로써 지연합의 개수가 증가함에 따라 인식률이 100%가 되는 것을 모의실험을 통하여 확인한다. 또한 본 논문에서는 음성의 프랙탈적인 특징이 화자인식에 사용될 수 있는지를 검토한다. 화자인식은 동질의 집단에서 13명의 성인남자의 목소리를 이용하여 단힌집합(closed-set)의 경우로 모의실험을 하였고, 기존의 특징으로는 선형예측계수(LPC)와 LPC-cepstrum을 사용하였다.

Abstract

In this paper, we propose a text-independent speaker identification system which has a classifier composed of two parts; to calculate the degree of likeness of each speech frame and to select the most probable speaker from the entire speech duration. The first part is realized using RBFN which is self-organized through learning and in the second part the speaker is determined using a combination of MAXNET and delayed summings. And we use features from linear speech production model and features from fractal geometry. Closed-set speaker identification experiments on 13 male homogeneous speakers show that the proposed techniques can achieve the identification ratio of 100% as the number of delays increases.

1 서론

화자 인식은 사람의 목소리로 화자의 신원을 파악해 내는 것이며, 사용하는 구문에 따라서, 화자의 등록과 인식에 같은 말을 사용하는 구문종속(text-dependent)적인 방법과 임의의 구문을 사용하여 화자의 등록과 인식을 할 수 있게 하는 구문독립(text-independent)적인 방법으로 나눌 수 있다. 일반적으로 구문종속적인 방법에서는 구문독립에서보다 높은 인식률을 보여주지만, 미리 정해진 말을 사용해야 한다는 불편함이 있다. 그 밖에, 모든 단어에 대하여 화자의 정보를 등록하고, 인식할 때에는 화자인식 시스템이 제시하는 구문을 화자가 발음하도록 하는 구문대기(text-prompt)의 방법도 있다[1,2]. 이 방법은 구문종속이나 구문독립적인 방법이 녹음기를 사용하여 화자인식 시스템을 속이는 것에 대하여 무방비 상태인데 비해, 녹음기의 사용을 불가능하게 할 수 있다는 장점을 가지고 있으며, 특히 화자검증의 경우에 유용하게 사용될 수 있다. 그러나 이 방법은 화자를 등록시키는

본 연구는 일주학술문화재단 연구비에 의해 수행되었음.

데에 상당히 많은 양의 데이터가 필요하다는 단점이 있다. 따라서 보안과 같은 목적으로 사용하지 않는다면 임의의 구문으로 등록과 인식을 할 수 있는 구문독립적인 방법으로 높은 인식률을 얻는 것이 가장 좋을 것이다. 화자인식 기법은 구문종속과 구문독립의 경우에 각각 다르게 발전해 왔다. 먼저 연구되었던 구문종속 화자인식에서는 스펙트로그램[3]이나 피치[4] 등의 주파수 특성들이 시간축 상에서 변하는 모습을 화자의 특징으로 하여 기준 패턴을 작성하고, 시험 패턴과의 유사도를 계산하여 신원을 판단하는 방법을 사용했다. 이 방법에서의 문제점은 같은 사람이 같은 말을 하더라도 말하는속도에 따라서 패턴 정합이 잘 이루어지지 않을 수 있다는 것이다. 따라서 이 방법에서는 시간축 상에서 음소들이 나타나는 시점을일치시키는 것이 중요한 문제가 되었으며 DTW(Dynamic Time Warping)등의 방법을 사용하여 성공적으로 화자인식을 수행하였다[5].

구문독립 화자인식에서 사용된 방법들은 다음과 같이 크게 4가지가 있다. 첫째는 피치나 선형예측계수 등의 구문종속적인 특징들을 긴 시간 동안에 평균하여 구문독립적인 특징 즉, 그 화자의 특성만을 나타내는 특징요소로 사용하는 것이다. 그러나 이 방법은 너무 많은 정보를 잃어버리게 되고 다른 사람의 특징을 계산했을 때 우연히 같은 값이 나올 수도 있으므로 믿을 만한 방법이라고 하기 어렵다. 평균 매칭법, 통계적 특징 평균법(statistical feature averaging)이라고도 불리며, 어떤 특징을 쓰느냐, 특징벡터 사이에 어떤 메트릭을 쓰느냐에 따라 여러 가지가 있을 수 있다. 둘째 방법은 앞의 구문 대기어서 사용됐던 방법과 유사한데 다만 말할 구문을 컴퓨터가 결정하지 않고 사용자가 임의로 선택한다는 점이 다르다. 즉, 컴퓨터는 사람의 말을 음소별로 분류해서 그 음소를 각 화자의 음소 모델과 비교하여 화자를 결정하는 것이다. 이것은 음소를 인식하는 데에서 잘못이 있을 수 있기 때문에 인식률은 '구문 대기'에 비해 많이 떨어진다[6].

셋째는 통계학적인 방법을 사용하는 것이다. 이것은 음성을 각 구간에 대해 특징을 추출한 다음에 각 구간의 특징이 개별적으로 화자의 특성을 대변한다고 보는 것이다. 요즘의 많은 연구들이 통계적인 방법을 사용하고 있으며[7,8,9,10], 이것은 화자를 모델링하는 확률 분포의 모습이 얼마나 미리 결정되어 있는지에 따라 매개변수(parameter)를 가지는 방법(parametric method)과 매개변수를 갖지 않는 방법(nonparametric method)으로 나눌 수 있다. 매개변수를 취하는 모델링의 예는 uni-modal Gaussian 또는 GMM 등이 있으며, 임의의 확률 분포를 모델링할 수 있는 후자가 전자보다 훨씬 좋은 결과를 나타낸다[7]. 이것은 음성 특징들의 분포가 단순하게 모델링되지 않음을 보여주는 것이다. 매개변수를 취하지 않는 방법에서는 특징벡터들의 분포를 몇 개의 매개변수로 결정되는 있는 확률 분포로 모델링하지 않고, 그 분포에 대해 최소한의 가정만 하여 보다 많은 자유도를 허용한다.

매개변수를 취하지 않는 모델링에는 벡터 양자화(VQ)[8]나 NN(Nearest Neighbor)[10] 등이 있다.

넷째는 신경회로망(Neural Network)을 사용하는 방법이다. 신경회로망은 각 사람마다 네트웍을 따로 두지 않고 특징 요소 공간에서 화자와 화자를 구분하는 경계가 되는 결정 함수(dicision function)를 학습하는 것이다. 일반적으로 신경회로망은 각각의 화자를 모델링하는 것에 비해 적은 수의 파라미터로 효율적인 학습을 시킬 수 있으며, 새로운 데이터를 추가시키더라도 이전에 학습된 것에 대한 정보가 남아있다는 장점이 있다. 그러나 신경회로망은 일반적으로 새로운 화자가 더해지면 입력 공간에서 결정 함수가 많이 달라지기 때문에 학습을 처음부터 다시 해야 한다는 것이 신경회로망을 이용하는 방법의 단점으로 꼽히고 있다. 신경회로망을 이용한 화자인식 연구는 현재 초기 연구단계로, 여러 가지 구조의 신경회로망이 연구되고 있다. 신경회로망의 학습속도를 빠르게 하기 위하여 이진트리 구조의 신경회로망이 제안되었고[11] 방사형기저함수망(RBFN)을 이용하여 화자 검증만을 행한 경우도 있다[12].

신경회로망을 이용한 화자인식에서 신경회로망은 분류기로 사용되며 입력은 음성으로부터 추출된 특징벡터가 된다. 음성은 수십 밀리초의 짧은 구간 동안을 기본 처리 단위로 하는데, 이 구간에서 나온 여러 특징들은 화자에 따라서 달라질 뿐만 아니라 같은 화자가 말할 때에도 화자의 기분과 발화상황에 따라서 크게 영향을 받는다[12]. 그러므로 한 구간에서의 특징들만으로는 화자를 완전히 분류해낼 수 없고 여러 구간에서의 특징들을

이용하여야 한다. 이것을 해결하기 위하여 시간지연 신경회로망(TDNN)을 이용하여 음성 특징들의 동역학적인 정보들을 학습시키기도 하였다[13]. 그러나 TDNN에서는 보통의 신경회로망을 이용한 방법과는 달리 앞뒤 몇 개의 구간에서 구한 특징들도 신경회로망의 입력이 되어야 하므로, 신경회로망의 규모가 수십 배로 커지며 이에 따라 학습이 어려워지게 된다. 또한, TDNN은 음성인식이나 구문종속인 경우에는 효과적일 수 있으나 구문독립일 경우에는 일반화능력이 떨어진다.

본 논문에서는 특징벡터들로부터 화자를 인식하기까지의 과정을 두 단계로 나누어, 각각의 특징벡터를 분류하는 신경회로망과 그것의 출력인 분류벡터들의 지연합으로부터 최종적인 화자를 결정하는 화자결정 회로망으로 구성하는 방법을 제안한다. 전자인 프레임분류 신경회로망은 특징벡터 공간에서 각 특징벡터가 어떤 화자의 특징과 가장 유사한지를 판별하여 분류하는 기능을 수행하며, 변형된 방사형기저함수망으로 구성한다. 후자의 화자결정 회로망은 분류벡터를 시간적으로 누적하여 최종결과를 내며 MAXNET[14]과 지연합을 이용하여 구성한다. 화자결정 회로망은 학습될 필요가 없다.

특징으로는 LPC, LPC-cepstrum, 프랙탈 차원 등을 사용하였으며 이 특징을 입력으로하여 각 프레임 단위로 신경회로망을 학습한다. 모의 실험을 통하여 다층인식자, 평균 매칭법 등과 성능을 비교 평가하고, 제안된 구조에서 특징벡터 및 여러가지 파라미터의 변동에 따른 성능 변화를 고찰한다.

2 제안된 화자인식시스템

그림 1은 제안된 구문독립 화자인식 시스템의 전체구조를 보여준다. 제안된 화자인식 시스템은 전처리기(preprocessor), 프레임 특징추출기(frame feature extractor), 프레임 분류 신경회로망(frame classification neural-net), 화자 결정회로망(speaker decision network)으로 구성되어 있다. 음성신호는 전처리 과정을 거친 다음에 여러 개의 프레임들로 재구성된다. 각각의 프레임들은 프레임 특징분류기에 지연 입력되어, 프레임 특징벡터(frame feature vector)가 계산된다. 프레임분류 신경회로망에서는 프레임 특징벡터가 각각의 화자에 속하

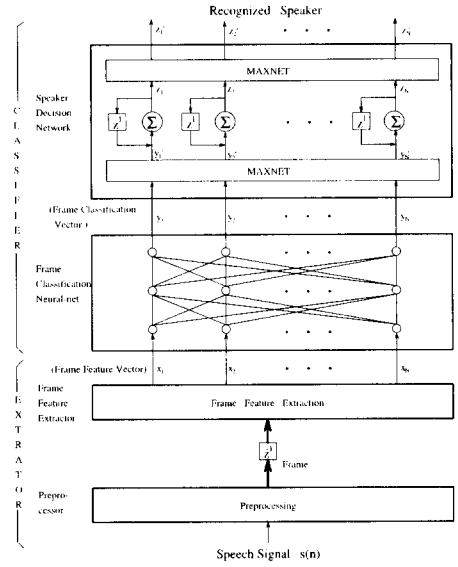


그림1: 구문독립 화자인식 시스템의 전체구조

는 정도를 출력으로 내보내며 이 출력을 프레임 분류벡터(frame classification vector)라고 부른다. 마지막으로 화자결정 회로망에서는 여러 개의 프레임 분류벡터들로부터 화자를 결정하게 된다.

2.1 전처리기

그림 2는 전처리기의 구조를 보여준다.

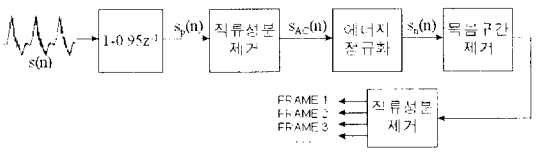


그림2: 전처리기

주어진 음성 신호는 먼저 $1 - 0.95\ Z^{-1}$ 에 의해 고주파 부분이 강조되어 녹음하는 과정에서 감소된 고주파 성분을 보상해 준다. 그리고 원래의 음성신호에는 직류성분이 없으므로, 마이크로 녹음되어 디지탈 신호로 변환되는 과정에서 생길 수 있는 직류성분을 제거해 준다. 그 다음에 전체 음성신호의 분산을 일정하게 하는 방법으로 신호의 파워를 일정하게 해 준다. 묵음 제거는 신호를 100 msec 길이의 겹치지 않는 구간들로 나누어서 각각의 표준편차를 계산한 다음에, 각 구간에 대하여

그 구간과 바로 앞뒤 구간의 표준편차의 가중합을 계산하여 그것이 임계값보다 작으면 그 구간 전체가 묵음으로 판정되도록 하였다. 그리고 묵음으로 판정되지 않은 구간들을 차례로 연결하여 1차원의 신호로 만든 다음에, 다시 한번 직류성분을 제거하고 프레임 형식으로 변환한다. 프레임의 길이는 256 샘플이고, 샘플링 주파수는 11.025 kHz이므로 한 프레임은 약 23 msec에 해당한다. 그리고 128 샘플마다 프레임을 취하여 각 프레임이 이웃하는 프레임들과 절반씩 중첩되도록 하였다.

2.2 특징추출기

본 논문에서 사용된 특징으로는 선형예측계수, 켑스트럼계수, 프랙탈 특징 등이 있으며, 음성신호의 프랙탈적인 특성이 화자인식에 유용한지를 알아보기 위하여 켑스트럼계수에 프랙탈 특징들을 조합하여 각각의 성능을 비교하였다 (3.3절). 다음 절에 이들 특징을 추출하는 방법과 특징벡터의 구성에 대해 기술한다.

2.2.1 선형예측계수 및 켑스트럼계수

선형예측계수(LPC)는 자기상관함수법(autocorrelation method)을 이용하여 구했고[15], 12차까지만 사용하였다. 켑스트럼은 계산 시간을 절약하기 위하여 LPC로부터 계산된 LPC-cepstrum을 사용하였으며, 식 (1)과 같은 방법으로 구할 수 있다[16].

$$c_n = a_n + \sum_{k=1}^{n-1} \left(\frac{k}{n}\right) c_k a_{n-k}, n = 1, \ldots, p \qquad (1)$$

여기에서 $\{c_i\}$는 LPC-cepstrum 계수이고, $\{a_i\}$는 선형예측계수이다. 본 논문에서는 LPC-cepstrum을 간단히 켑스트럼이라고 하겠다.

2.2.2 프랙탈 차원

인간의 발성기관은 비선형 혼돈역학 특성을 가지고 있는 것으로 알려져 있으며[17] 이것으로부터 발생되는 음성에는 많은 비선형적인 요소들이 있다. 비선형성의 가장 큰 원인은 말하고자 하는 내용 자체의 변동 때문이며, 이것은 음성을 짧은 구간들로 나눔으로써 상당부분 해결가능하다. 하지만, 구분적선형화를 거친 후에도 비선형적 요소들은 남아있어서 이것들은 음성인식이나 화자인식의 성능을 저하시킨다. 프랙탈 기하학은 유클리드 기하학에 비하여 자연에서 얻어진 복잡한 형태의 데이터에 대한 설명과 모델링이 용이하기 때문에 음성신호와 같이 복잡하고 잘 알려지지 않은 신호를 다루는 데에 적합하다고 할 수 있다[18]. 프랙탈은 신호에서 보이는 매우 복잡한 구조로 정의될 수 있으며, 복잡한 정도는 프랙탈 차원으로 나타나므로 이 프랙탈 차원이 화자를 나타내는 특징이 될 가능성이 있다. 따라서 본 논문에서는 음성으로부터 혼돈역학의 특성을 반영한 프랙탈 차원 등이 화자인식에 미치는 영향을 실험적으로 분석하고자 한다.

프랙탈 차원을 얻는 방법은 여러 가지가 있으나 여기에서는 주파수 영역을 이용하는 파워 스펙트럼 방법(power spectrum method)을 사용하였다[19]. 파워 스펙트럼 방법에서는 신호를 프랙탈--브라운(Fractal-Brownian) 운동으로 모델링한다. 음성의 스펙트럼을 $P(f)$라고 하면 (f는 주파수이다) 프랙탈-브라운 모델에서는 파워 스펙트럼이 주파수에 지수승으로 반비례하며, 이 지수값 β로부터 프랙탈 차원을 구할 수 있다[20].

$$p(f) \approx \frac{1}{f^{\beta}} \quad or \quad \beta = -\frac{\log p(f)}{\log f} \qquad (2)$$

$$\beta = 5 - 2D \qquad (3)$$

여기에서 D가 프랙탈 차원이며, 음성신호의 차원이기 때문에 $1<D<2$를 만족한다. DFT를 이용하여 음성신호의 파워 스펙트럼을 구하면 β는 대수눈금의 주파수-스펙트럼 그래프에서 기울기로 나타난다. 따라서, 주파수 스펙트럼 그래프를 $y=-\beta x+b$로 나타내고 기울기를 최소자승법으로 추정하면, 프랙탈 차원 β는 식 (4)와 같이 구할 수 있다.

$$\beta = \frac{N\sum_{i=1}^{N} y_i x_i - \sum_{i=1}^{N} y_i \sum_{i=1}^{N} x_i}{\sum_{i=1}^{N} x^2{}_i - \left(\sum_{i=1}^{N} x_i\right)^2} \qquad (4)$$

2.2.3 Lacunarity

Lacunarity는 프랙탈 차원을 보완하는 중요한 특징으로서 분포의 차이로 정의될 수 있다. 즉, 이 값은 식 에서 추정된 1차식과 원래의 값과의 차의 제곱의 평균으로 정의된다. Lacunarity L은

$$L=\frac{1}{N}\sum_{i=1}^{N}\left(y_i+\beta x_i-b\right)^2 \qquad (5)$$

이며, 여기에서 b는 식 (4)에서와 같이 최소 자승법을 이용하면

$$b=\frac{\sum y_i\sum x_i^2-\sum x_i y_i\sum x_i}{\sum x_i^2-\left(\sum x_i\right)^2} \qquad (6)$$

와 같이 구할 수 있다. 이 방법의 경우 b는 식 에서 사용된 값들을 써서 연산을 하므로 계산량이 많지 않으나, L을 구하는 데에는 많은 계산이 필요하다.

2.2.4 특징벡터의 구성

다음과 같이 4가지의 방법으로 특징벡터를 구성하여 실험에 사용하였다.

- 특징벡터Ⅰ:12차의 선형예측계수:[a_1 a_2 a_3,... a_{12}]
- 특징벡터Ⅱ:12차의 켑스트럼계수:[c_1 c_2 c_3,... c_{12}]
- 특징벡터Ⅲ:10차의 켑스트럼계수 + 음성의 프랙탈 차원 + 음성의 lacunarity:[c_1 c_2 ... c_{10} d_s l_s]
- 특징벡터Ⅳ:10차의 켑스트럼계수 + 예측오차의 프랙탈 차원 + 예측오차의 lacunarity:[c_1 c_2 ... c_{10} d_e l_e]

여기에서 {a_i} 와 {c_i}는 각각 선형예측계수와 켑스트럼계수이고, d_s, l_s와 d_e, l_e는 각각 음성신호와 선형예측오차로부터 구한 프랙탈 차원과 lacunarity이다. 이들 특징요소들은 각 요소별로 구한 최대값과 최소값을 이용해 0과 1사이의 값으로 선형변환된 이후에 분류기의 입력으로 사용되었다.

특징벡터Ⅰ은 12차의 선형예측계수이고, 특징벡터Ⅱ는 12차의 켑스트럼계수로서 선형예측계수를 사용한 것과의 인식률의 차이를 알아보기 위한 것이다. 그 다음에는 켑스트럼계수에 프랙탈적인 특징인 프랙탈 차원과 lacunarity를 포함시켜 특징벡터를 구성하였다. 그러나 특징요소의 개수를 같게 하여야 공정한 비교가 될 것이므로, 특징벡터Ⅲ, Ⅳ에서는 켑스트럼계수를 10개만 사용하여 다른 특징벡터와의 성능비교가 가능하게 하였다. 특징벡터Ⅲ은 음성신호 자체의 프랙탈 특징을 특징요소로 사용한 것이고, 특징벡터Ⅳ는 음성신호를 선형발성모델로 예측하였을 때의 오차신호로부터 구한 프랙탈 특징을 특징요소로 사용한 것이다.

선형발성모델에서는 음성을 성대라는 선형시스템에 입력이 들어갔을 때에 나오는 출력으로 생각한다[15]. 이 모델에서는 성대가 전극점형으로 나타나므로 음성신호에 선형예측기법을 적용하여 성대의 특징계수들을 쉽게 구할 수 있으며, 이 때 발생하는 예측오차가 곧 성대의 입력이 된다. 성대의 모양과 성대의 입력신호는 말하는 단어나 화자 등 여러가지 조건에 따라 변이가 있으며, 단어나 화자 중의 어느 한가지로부터만 영향을 받는 요소는 없다. 기존의 LPC나 켑스트럼은 성대의 모양으로부터 구한 특징인데, 본 논문에서는 성대의 입력에 담겨있는 화자의 정보를 이용하기 위하여 특징벡터Ⅳ를 시도해보았다. 특징벡터Ⅳ는 성대 입력의 프랙탈 차원을 특징요소로 사용한 것이다. 단순히 음성신호의 프랙탈 특징을 사용한 특징벡터Ⅲ보다 선형예측모델에서 사용되지 않은 정보를 이용한 특징벡터Ⅳ가 더 좋은 결과를 낼 것으로 기대할 수 있다.

2.3 프레임분류 신경회로망

프레임분류 신경회로망은 특징추출기에서 얻어진 프레임 특징벡터가 어떤 화자에 가장 유사한지를 판별하여 분류하는 기능을 수행한다. 여기에서는 방사형기저함수망[12,21]을 변형하여 구성하였다. 이 신경회로망의 구성은 학습에 의해 이루어지며, 각 프레임 별로 학습이 이루어진다. 즉, 각 음성프레임의 특징 벡터를 입력으로 하고 출력은 그 음성 프레임이 각 화자의 클래스에 속할 가능성을 정량적으로 나타내는 값이 된다.

2.3.1 방사형기저함수망 구조

본 논문에서 프레임분류 신경회로망으로 사용한 방사형기저함수망의 구조는 그림 3과 같다. 방사형기저함수망을 이용한 프레임 분류기입력은 k번째 프레임에서 추출된 프레임 특징벡터 $\vec{x}(k)$이며 출력은 k번째 프레임 분류벡터 $\vec{y}(k)$ 이다. 특징벡터는 모든 노드에 입력되며, 노드들은 N개의 분할(partition)로 나뉘어진다. 여기에서 N은 등록된 화자의 수이다. j번째 네트웍에 속해있는 i번째 노드의 출력을 ϕ_{ji}라고 하면, 입력 $\vec{x}(k)$에 대한 출력

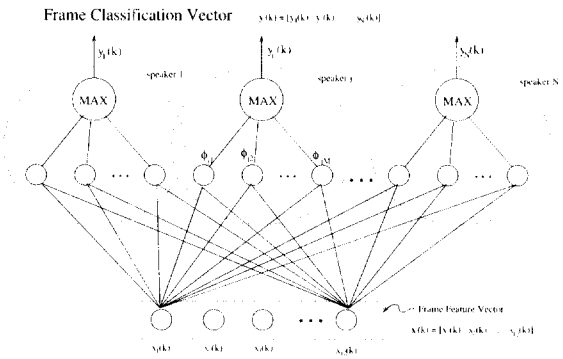


그림 3. 방사형기저 함수망을 이용한 프레임 분류기

$\phi_{ji}(\vec{x})$ 은 식 (7)로 주어진다. 식을 간단히 하기 위해 $\vec{x}(k)$ 를 $\vec{x}$로 나타내었다.

$$\phi_{ji}(\vec{x}) = \exp\left(\frac{-\left\|\vec{x}-\vec{w}_{ji}\right\|^2}{\sigma^2}\right) \qquad (7)$$

여기에서 $\vec{w}_{ji}$는 노드의 중심이며, 각 노드마다 입력과 같은 크기의 벡터를 하나씩 갖는다. 또, σ^2은 노드의 모양을 결정하는 변수로서 그 값이 작을수록 뾰족한 노드가 된다. 본 논문에서는 노드의 모양을 모두 같게 하였다. 이 노드의 출력은 0에서 1 사이의 값으로서 입력 벡터와 노드 중심과의 유클리드 거리가 작을수록 커지게 된다.

각 네트워의 출력은 노드 출력들의 최대값이고, 전체 N개의 네트워크으로 이루어진 이 함수망의 최종 출력 $\vec{y}(k)$ 는 식 (8)과 같이 된다.

$$\vec{y}(k) = \left[y_1(k)y_2(k)...y_N(k)\right]$$
$$y_j = \max\left\{\phi_{j1}(\vec{x}),\phi_{j2}(\vec{x}),...,\phi_{jM_j}(\vec{x})\right\} \qquad (8)$$

2.3.2 방사형 기저함수망의 학습

본 논문에서 제안된 방사형기저함수망은 아주 간단한 구조로서 각 화자마다 독립된 네트워크을 갖고 각각은 여러 개의 노드로 구성되어 있으며 각 노드는 중심벡터만을 가진다. 각 네트워크이 가지는 노드의 수는 변할 수 있으며 학습을 통해서 조정된다. 본 논문에서는 노드를 추가시킬 수만 있게 하여 알고리듬을 간단히 하였다. 즉, 처음에는 0개의 노드로 시작하며, 학습이 진행됨에 따라 조건이 만족되면 노드가 생성된다.

이 함수망은 각각의 화자에 대해 독립적으로 학습된다. 이를 테면, j번째 화자를 학습시킨다고 하자. 그러면 학습시키고자 하는 화자의 프레임 특징벡터 ($\vec{x}(k)$)를 입력으로 하여 j번째 네트워크의 출력($y_j(k)$)을 정해진 임계값과 비교한다. 그래서 네트워크의 출력이 임계값보다 크면, 그 출력을 낸 노드(i번째 노드라고 하면)의 중심벡터를 식 (9)와 같이 갱신시킨다.

$$\vec{w}_{ji}^{(p+1)} = \vec{w}_{ji}^{(p)} + \frac{1}{p+1}(\vec{x}(k) - \vec{w}_{ji}^{(p)}) \qquad (9)$$

여기에서 p는 그 노드가 갱신된 횟수이고, 한 번 갱신될 때마다 1씩 증가한다. 그리고 네트워크의 출력이 임계값보다 작을 때에는 새로운 입력을 중심으로 하는 노드를 생성하여 그 네트워크에 추가시킨다. 즉, m번째 노드가 생성되었다고 하면 $\vec{w}_{jm}^{(1)} = \vec{x}(k)$이 된다. 단, 노드가 하나도 없을 때에는 네트워크의 출력을 0으로 정의한다. 이러한 과정을 모든 화자의 특징벡터들에 대하여 반복하면 학습이 끝난다.

제안된 방사형기저함수망의 파라미터는 각 노드마다 한 개의 중심벡터와 그 노드가 몇 번 학습되었는지를 저장하는 변수 한 개이다. 가중치는 사용되지 않았다. 화자당 파라미터의 수는 생성된 노드의 개수에 비례하며, 비례상수는 입력공간의 차원에 1을 더한 값이다(여기에서는 13이 된다). σ^2에 따라서 기저함수의 모양이 달라지며, 이에 따라 생성된 노드의 개수도 달라진다. σ^2이 작으면 기저함수가 뾰족한 모양이 되므로 같은 데이터를 학습하는 데에 더 많은 노드가 필요할 것이라고 짐작할 수 있다. 일반적으로 노드가 많아지면 인식률도 대체로 높아진다. 그러나 하드웨어로 구현할 경우 각 네트워크이 가질 수 있는 노드의 개수에는 한계가 있을 것이므로, 노드의 개수가 많아지는 것은 바람직하지 못하다. 뿐만 아니라, 일반화 성능을 높이기 위해서는 가능하면 적은 수의 노드로 학습을 시키는 것이 바람직하다. 기저함수의 모양에 따른 노드의 개수 및 인식률의 변화에 대해서는 3.4.2절에서 고찰하였다.

2.4 화자 결정 회로망

프레임분류 신경회로망에서는 입력된 프레임 특

징벡터가 각각의 화자에 대해 어느 정도의 상관관계를 가지는지를 프레임 분류벡터로 나타내 준다. 그러나 이 벡터의 각 원소들은 0과 1사이의 연속적인 값이고, 무엇보다도 한 프레임에 대한 정보만을 나타내기 때문에 여러 프레임에서 나온 프레임 분류벡터들을 종합하여 최종적인 판단을 내리는 화자결정 회로망이 필요하다.

화자결정 회로망에서는 프레임 특징벡터로부터 구한 분류벡터들을 입력으로 받아 화자를 결정한다. 화자결정 회로망의 구조는 앞에서 그림1에 제시되었다. 입력은 프레임분류 신경회로망의 출력인 프레임 분류벡터이고, 출력은 N개 원소 중 1개만 0이 아닌 벡터이다. 프레임 분류벡터 $\hat{y}(k)$들은 먼저 MAXNET[14]을 통해, 가장 큰 값을 제외한 다른 모든 값들이 0이 된 $\hat{y}^*(k)$ 가 된다. 이 벡터들은 모든 음성 프레임에 대해 누적되며, 이 누적합들은 다시 MAXNET을 거쳐 누적합이 가장 큰 노드만 0이 아닌 값을 출력하고 나머지 화자노드는 0을 출력함으로써 화자를 결정하게 된다. 누적합의 반복회수가 클수록, 즉 더 많은 음성 프레임이 입력될수록 화자 인식률이 증가된다.

3 모의실험 및 성능평가

이 절에서는 본 논문에서 제안한 방법을 사용하여 구문독립 화자인식을 모의실험한 결과를 제시한다. 모의실험방법을 설명한 다음에, 제안된 분류기와 다른 분류기와의 성능비교와 앞에서 구성한 4가지의 특징벡터에 따른 성능비교를 한다. 특징벡터의 성능비교는 구문종속과 구문독립에서 하였다. 그리고 학습 데이터의 수와 망 규모에 따른 분류기의 성능을 알아본다.

3.1 모의 실험 방법

3.1.1 화자 데이터

실험에 사용된 화자는 13명이며 모두 한국인이고 언어는 국어만 사용했다. 화자는 모두 20대의 성인 남자들이고 직업은 대학원생이다. 녹음은 특별한 잡음 제거 시설이 없는 일반 실험실에서 11.025 kHz, 16 bit로 하였고, 각 화자가 녹음할 때의 잡음 정도는 대체로 비슷한 편이었다. 말하는 내용은 임의의 문장으로 하여 10초 이상 말하기를 13번 하였으며(각각이 하나의 소리파일로 녹음되었다), 13번 녹음하는 동안 목소리가 모두 일정한 톤이나 어투가 되지 않게 하려고 노력하였다. 각 화자마다 13번 말하였는데 그 중에서 5번은 분류기를 학습하는 데에 사용하였고 나머지 8번은 테스트하는 데에 사용하여, 전체적으로 테스트를 하는 데에 사용된 소리파일의 개수는 104개이다. 이 데이터 파일들은 Microsoft Windows 3.1의 'wav' 파일 형식으로 되어 있다.

화자마다 약 10초 동안의 음성을 녹음하기는 했지만 그 10초간의 녹음 중에는 음성이 들어있지 않은 묵음구간도 있기 때문에 모든 소리파일들이 실제로 똑같은 길이의 음성을 담고 있는 것은 아니다. 그리고 어떤 화자에 대해서는 평균적으로 더 긴 시간동안 녹음한 경우도 있으므로, 만일 104개의 소리파일에서 나오는 프레임 특징벡터들을 모두 다 사용한다면 특정 화자에 대해서 편중된 결과를 가져오는 분류기가 더 우수한 성능을 보일 수도 있을 것이다. 실제로 본 논문에서 제안된 것과 같은 방법으로 자기조직하는 RBFN의 경우에는 특정 화자에 대하여 학습 데이터와 테스트 데이터 개수가 모두 다른 화자의 데이터보다 많다면, 그 화자에 대해서 더 많은 노드를 가지는 분류기가 그렇지 않은 것에 비해서 더 높은 인식률을 나타낼 것이라고 예상할 수 있다. 그러므로 본 논문의 모든 모의실험에서는 테스트할 때 각 화자 데이터의 비율을 균일하게 해 주기 위해서 모든 테스트용 데이터 파일에서 같은 개수의 프레임만을 뽑아내어 사용하였다. 그렇게 하기 위해서는 먼저 모든 데이터 파일에서 프레임 특징벡터를 뽑아내고, 그 다음에 각 파일에서 구해진 특징벡터의 개수를 조사해서 그 중의 최소값을 구해 그 개수만큼의 특징벡터만을 실험에 사용하는 방법을 썼다.

3.1.2 성능 평가

이후의 절에서 나오는 모의실험 결과들은 테스트 음성의 길이를 변화시켜 가면서 인식률을 계산한 것이다. 그런데 소리파일들에는 묵음이 많이 있을 수도 있고 적게 있을 수도 있으므로 소리파일에서 직접 취한 데이터의 길이와 음성의 길이는 다를 수 있다. 따라서 본 논문에서는 소리파일들에서 묵음을 제거한 다음에, 그 음성을 프레임 단위로 나누어 그 프레임의 개수로써 음성의 길이를

재계산하는 방법을 사용하였다. 이와 같이 음성의 길이를 재계산했을 때, 10초간 녹음한 소리파일이라 하더라도 그 중에 묵음이 절반 정도는 차지하여 순수한 음성의 길이는 약 5초 남짓 밖에 되지 않는 것으로 나타났다.

이후의 성능평가 결과에서 테스트 음성의 길이는 0.1초에서부터 약 5초까지를 대수간격으로 20등분하여 사용하였다. 그래서 각 테스트 데이터 중에서 원하는 테스트 시간(test speech duration)에 해당하는 개수(T)만큼의 연속된 프레임 특징벡터들(이것을 segment라고 하자)을 1초[1]마다 취하여 테스트 세그먼트를 다량확보하고, 이들로 화자인식을 수행한 결과의 평균 인식률을 계산하여 그것을 그래프로 나타내었다. 즉, 어떤 소리파일에서 나온 특징벡터가 $\{\vec{x}_1, \vec{x}_2, \vec{x}_3, \ldots\}$ 이라면, 맨 처음의 두 세그먼트는 다음과 같이 된다.

$$\overbrace{\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_T}^{Segment1}, \vec{x}_{T+1}, \vec{x}_{T+2}, \ldots$$

$$\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_S, \overbrace{\vec{x}_{S+1}, \ldots \vec{x}_{T+S}}^{Segment2}, \vec{x}_{T+S+1}, \ldots$$

S는 1초에 해당하는 프레임의 개수로서 86이고, T는, 예를 들어 한 세그먼트의 길이가 5초라면 431에 해당한다. T개의 벡터들의 한 묶음인 세그먼트는 각각이 서로 별개의 것으로 다루어진다. 각 세그먼트에서 식별된 화자는 그 데이터의 원래 화자와 비교되어서, 옳게 식별된 세그먼트 개수와 테스트된 전체 개수의 비율이 그 때의 인식률이 된다.

3.2 분류기에 따른 성능평가

비교한 분류기는 (a) 제안된 RBFN을 사용한 것, (b) MLP를 이용한 것, (c) 단순히 특징 벡터의 평균만으로 비교하는 방법(statistical feature averaging)이다. RBFN의 각 노드의 σ^2은 0.2로 하였고, 새로운 노드를 추가할 때의 임계값은 0.14로 하였다.[2] MLP[22]는 은닉층이 1개가 있는 것을 사용하였고, 입력노드는 12개, 출력노드는 13개, 은닉노드는 13개이며, 모든 은닉노드와 출력노드는 바이어스를 갖는다. MLP의 학습은 adaptive learning rate와 모멘트 방법을 사용하였으며 학습회수는 2,000 epoch로 하였다. 특징벡터는 인식률이 가장 좋은 IV번을 사용하였고, 학습패턴은 각 소리파일에서 적당한 개수씩을 균일하게 뽑아내어 구성하였으며, 테스트패턴은 학습패턴에 사용되지 않은 소리파일들에서 구성하였다. 학습에 사용된 소리파일은 화자마다 5개씩인데, 그 하나하나에 대해서 23개씩의 특징벡터만을 무작위로 뽑고 또 모든 화자에 대해서 같은 방법으로 하여 전체적으로 1,500개 정도의 특징벡터로 구성된 학습 패턴이 생기도록 하였다. 단, 위의 세 가지 방법에서 사용된 학습 패턴은 서로 다를 수 있다. 학습 패턴의 크기를 이렇게 작게 한 것은 RBFN과 비교했을 때 상대적으로 학습 용량이 작고 대신 일반화 능력이 뛰어난 MLP가 조금이라도 유리해지도록 하기 위한 것이었다.

그림 4는 이 세 가지 경우에 대한 인식률을 나타낸다. 위의 그래프에서 보이는 인식률은 다음 절

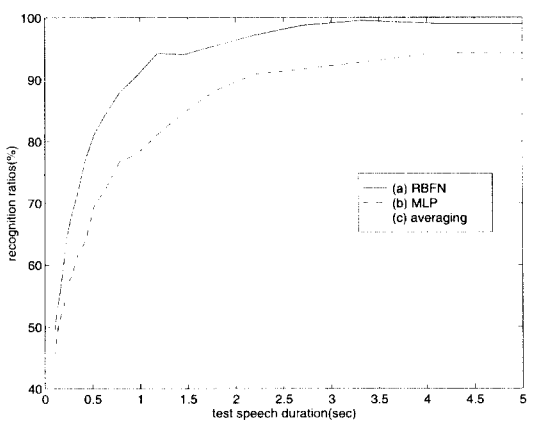


그림4: 분류기에 따른 인식률 (a)RBNFN (b)MLP (c)평균 매칭법

에서 보이는 것보다 낮은 것을 알 수 있다. 이것은 학습패턴을 작게 하였기 때문이다. 실험 결과를 보면, 가장 간단한 방법인 평균 매칭법이 예상대로 가장 낮은 인식률을 보여준 것을 알 수 있다. 그러나 더 유리한 조건을 만들었음에도 불구하고 MLP를 이용한 방법이 RBFN을 이용한 방법보다 인식률이 더 나쁜 것을 볼 수 있다. 그리고 위의 실험에 사용된 학습패턴은 1,500개 정도 밖에 되지

[1] 실제로는 1초에 해당하는 프레임의 개수이다.
[2] 이 값은 생성되는 노드의 개수 및 인식률에 미치는 영향이 거의 없다.

않았지만, (a)와 (b)의 경우 인식률이 95% 정도까지 올라가는 것을 통해서 본 논문에서 제안된 방법이 어느정도의 일반화 능력이 있음을 알 수 있다.

3.3 특징벡터에 따른 성능평가

이 절에서는 앞에서 제시한 4가지 특징벡터에 따른 성능을 비교하였다. 먼저 특징벡터들만의 성능을 비교하기 위하여 DTW[23]를 이용하여 구문종속 화자인식을 행하였고, 이때에는 구문종속실험을 위한 별도의 음성데이터를 사용하였다. 구문독립실험에서는 제안된 RBFN 분류기를 이용하여 성능을 비교하였다. 또한, 제안된 RBFN 화자인식 시스템에서 화자들이 모두 같은 단어들을 사용할 때의 인식결과를 알아보기 위하여 구문종속실험에서 사용하였던 음성데이터들을 이용하여 구문독립 화자인식실험을 수행해보았다.

3.3.1 구문종속 화자인식

여기에 사용한 음성데이터는 0('영')부터 10('십')까지의 11개의 숫자음을 각각 8번씩 말한 것이다. 화자는 9명으로 하였다. 9명의 화자 모두에 대하여, 첫번째 말한 것을 기준패턴으로 하고 나머지 7개의 음성을 시험패턴으로하여 화자인식률을 구하고, 두번째 음성에 대해서도 이와같이 하여 모두 8번 반복하였다. 그리고 11개의 단어에 대한 인식률의 평균을 취하여 그 특징벡터의 인식률로 삼았다.

표1:구문종속에서의 특징백터별 오인식률

Ⅰ(LPC)	Ⅱ(켑스트럼)	Ⅲ(프래탈1)	Ⅳ(프래탈2)
18.0%	7.3%	8.5%	8.4%

표 1에 그 결과를 나타내었다. 여기에서는 단순 유클리드 거리를 사용하였으므로 위의 인식률이 최적화된 성능은 아니라고 할 수 있지만, 특징벡터들간의 대체적인 성능비교는 가능하다고 본다. 위의 결과로부터 프랙탈 특징이 구문종속적인 방법에서는 변별력을 감소시킨다는 것을 알 수 있다. 문종속에서의 특징벡터별 오인식률구문독립 화자인식 여기에서는 앞에서 설명한 구문독립용 음성데이터를 사용하였으며, 학습용 파일과 테스트용 파일에서 화자당 동일한 개수의 특징벡터만을 취하여 학습과 테스트에 사용하였다. 이는 인식률을 계산할 때 특별한 화자에 대한 치우침이 없도록 하기 위해서이다. RBFN에서 새로운 노드를 생성하는 임계값은 0.14로 하였으며, 은 인식률에 영향을 주므로 다음과 같이 두가지 방법으로 실험하였다. 첫번째 방법은 각 특징벡터에 대해서, 화자당 평균 노드의 수가 60보다 작으면서 60에 최대한 가깝게 하여 네트웍의 크기를 일정하게 하는 것이다. 두번째 방법은 노드의 모양과 관련있는 것이 각 특징요소의 분포도이므로, 각 특징요소의 분포를 평균 0, 분산 1로 맞추어주는 것이다. 이것을 위해 각 화자마다 학습패턴을 통계적으로 분석하여 각 화자별 평균과 분산을 저장해 두었다가, 테스트할 때에는 각각의 화자 네트웍에서 화자별 평균과 분산을 이용하여 입력을 정규화하는 방법을 사용하였다. 두번째 방법은 실제 응용에 사용되기에는 어려운 점이 있지만, 이렇게 함으로써 생성되는 노드의 개수를 줄일 수 있음이 확인되었다.

그림 5는 첫번째 방법의 결과를 보여준다. 구문독립에서 RBFN 규모를 일정하게 했을 때 LPC(Ⅰ번)는 다른 것에 비해 확실히 낮은 성능을 보여주며, 나머지 세개는 거의 비슷한 인식률을 보여준다. 하지만, 이번에는 특징벡터Ⅳ가 Ⅱ나 Ⅲ에 비해 약간 더 좋은 성능을 보여준다.

그림 6은 두번째 방법으로 했을 때의 결과이다. 구문독립에서 특징요소들의 분포를 일정하게 했을 때 이때에도 대체적인 경향은 변하지 않는다. 입력값들의 분포를 일정하게 하였는데도 오히려 인식률이 높아지지 않았는데, 이것은 그 전에도 이미 어느정도 정규화가 이루어진 상태였기 때문이다. 그 때에는 특징요소별 대략의 최대 최소값을 화자와 상관없이 구했고, 이번에는 각 화자별로 정규화를 했다는 차이가 있다.

3.3.3 같은 단어집합을 사용한 구문독립 화자인식

이 절에서는 구문독립 화자인식 실험을 행하되, 모든 화자가 동일한 단어들을 사용할 때의 결과를 알아보기 위한 실험을 해보았다. 음성데이터는 앞의 구문종속에서와 같이 9명의 화자가 숫자음 11단어를 발음한 것을 사용하였고, 이번에는 10번씩 말하여 5번은 학습에 이용하고 나머지 5번으로 테스트를 하였다. 그림7은 이때의 특징벡터별 인식률을 나타낸다. 같은 단어집합을 사용했을 때 구문종

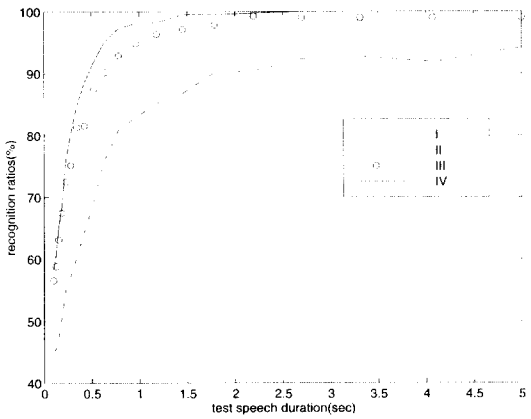


그림5. 구문독립에서 RBFN규모를 일정하게 했을때

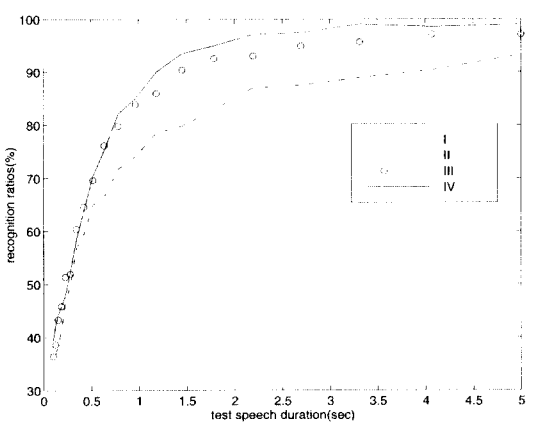


그림6. 구문독립에서 특징요소들의 분포를 일정하게 했을때

속적인 방법을 사용하지 않고서도 인식률이 상당히 높게 나왔는데, 이것은 같은 단어를 발음했기 때문이기도 하지만, 무엇보다도 대상 화자의 수가 많지 않기 때문이다. 이 때에는 특징벡터Ⅲ번의 성능이 상대적으로 낮게 나왔다.

3.3.4 특징벡터에 대한 분석

특징벡터들의 성능을 비교한 이 실험에서 구문종속이든 구문독립이든 켑스트럼을 사용한 것의 성능이 LPC를 사용한 것보다 항상 좋게 나왔다. 또한, 켑스트럼계수에서 2개를 프랙탈 특징으로 대치한 것의 경우에는 음성의 프랙탈 특징을 그대로 사용할 경우에는 성능의 향상이 없고 오히려 구문이 일정한 경우에는 더 성능이 나빠졌다. 그러나

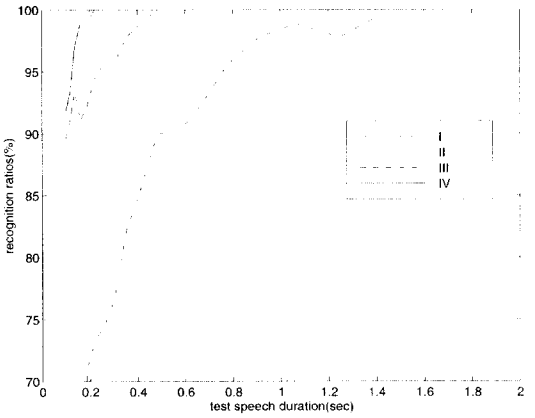


그림7. 같은 단어집합을 사용했을때

예측오차, 즉 성대의 입력신호에서 구한 프랙탈 특징을 사용할 경우에는 구문독립의 경우에 약간의 성능향상이 있을 수 있는 것으로 밝혀졌다. 하지만 그 차이가 크지 않고, 사용하는 분류기나 다양한 실험환경에 따라서도 조금씩 달라질 수 있기 때문에 프랙탈 특징을 이용한 화자인식에 대해서는 더 폭넓은 연구가 요구된다.

3.4 분류기의 성능평가

3.4.1 학습데이터 수에 따른 성능평가

이 절에서는 앞 절에서 가장 좋은 성능을 보이는 RBFN의 학습 능력을 알아보기 위해서 학습 데이터의 개수를 변화시켜 가면서 실험하였다. 사용한 특징벡터는 Ⅳ번이며, 여기서도 각 노드의 σ^2은 0.2로 하였으며 새로운 노드를 생성하는 임계값은 0.14로 하였다. 이 실험에서 학습 데이터의 개수를 변화시킨 방법은 다음과 같다. 앞에서 말한 것과 같이 학습에 사용하는 소리파일의 개수는 모두 65(=13 × 5)개이고 각각이 가지고 있는 데이터의 개수는 모두 다르다. 그래서 각 소리파일로부터 일정한 개수의 특징벡터들을 추출하여 전체의 학습 데이터의 개수가 1,500개에서 29,250개가 되도록 하였으며, 추출하는 방법은 무작위로 하였다. 29,250은 각 소리파일들로부터 같은 개수의 특징벡터들을 추출하여 얻을 수 있는 가장 큰 개수이다. 모두 8가지의 데이터 개수에 대해서 실험을 하였고, 여기에는 그 중 4가지의 경우만 나타내었다.

그림 8은 그 결과를 보여준다. 각각의 한 화자당

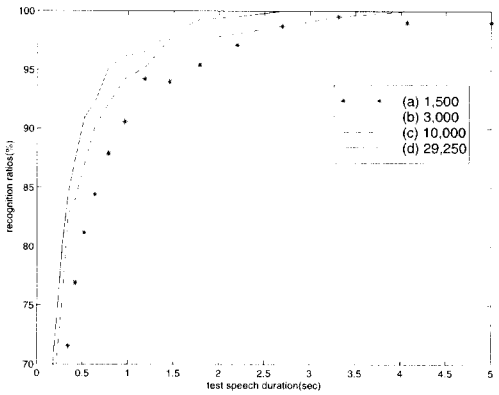


그림 8. 학습데이터 수에 따른 인식률 (a)1,500 (b)3,000 (c)10,000 (d)29,250

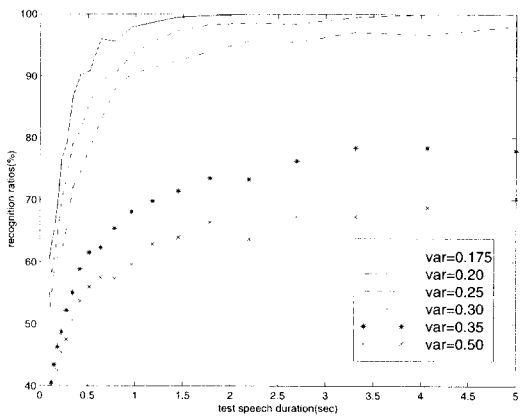


그림 9. 망 규모에 따른 인식률(var는 σ^2 를 의미한다)

표 2. 화자 한 명당 평균 노드의 개수

σ^2	0.50	0.35	0.30	0.25	0.20	0.175
노드수	1.0	3.1	7.6	20.5	66.9	134.8

데이터 개수는 (a) 115 (b) 230 (c) 770 (d) 2,250 이다. 예상한 대로 학습 데이터가 많아질수록 인식률이 대체적으로 높아지는 경향이 있음을 볼 수 있다. 그래프에서 (a)와 (b)는 4초 이후에 99.0%의 인식률을 보여주고, (c)와 (d)는 각각 4초 이후와 2.7초 이후에 100.0%의 인식률을 보여준다. 테스트 시간이 길지 않을 때에는 학습데이터의 수에 따라 인식률에 큰 차이를 보여주지만, 테스트시간이 충분히 길어지면 학습데이터가 적더라도 높은 인식률을 보여줌을 알 수 있다. 실험했던 8가지의 경우 모두 4초 이후엔 99.0%이상의 인식률을 보여주었는데, 이것은 앞에서의 평균 매칭법으로는 얻어질 수 없는 값이다.

3.4.2 망 규모에 따른 성능평가

이 절에서는 본 논문에서 제안한 형태의 RBFN의 노드의 개수를 변화시키면서 인식률을 비교해 보았다. 그러나 RBFN이 생성하는 노드의 개수를 직접 제어할 수 있는 방법이 없으므로 노드의 폭을 변화시킴으로써 결과적으로는 전체 네트워의 규모가 변화되도록 하였다. 여기서에도 새로운 노드를 생성하는 임계값은 0.14로 하였고, 사용한 특징벡터는 Ⅳ번이며, 학습패턴은 65개의 소리파일에서 사용할 수 있는 모든 특징벡터들로 구성하였다. σ^2은 0.175에서부터 0.5까지 변화시켰으며, 그 때에 생성된 화자 한 명당 노드의 평균 개수는 표 2와 같다.

그림 9는 각각의 인식률을 보여준다. 이 그래프로부터 σ^2의 적정값은 0.2인 것을 알 수 있다. 왜냐하면, 0.2보다 더 작은 값일 때는 인식률이 더 높아지지 않는 데다가 생성되는 노드의 수는 2배로 늘어나기 때문이다. σ^2이 0.5와 0.35일 때에는 화자당 노드의 수가 3개 이하인데 이 때에는 앞에서의 평균 매칭법에 의한 결과보다도 더 낮은 인식률을 보여주며, σ^2이 0.3이 되어 화자당 평균 노드수가 7.6이 되었을 때부터 인식률이 높아지는 것을 볼 수 있다. 이것으로부터 구문독립 화자인식에서 각 화자의 음성특징들이 나타내는 확률분포는 단순한 모양의 함수로써 쉽게 모델링되지 않을 것임을 알 수 있다.

3.5 실험결과 고찰

이 절에서는 본 논문에서 제안한 RBFN 분류기를, MLP를 이용한 분류기 및 평균 매칭법과 구문독립 화자인식의 성능 면에서 비교하였고, 그 중 RBFN 분류기가 상당히 높은 인식률을 보여줌을 확인했다. 특징벡터의 일부를 본 논문에서 제안한 프랙탈 특징들로 대치하였을 때 인식률이 더 높아지는 경우를 통하여, 프랙탈 기하학이 화자인식의 연구에 어느 정도 유용하게 사용될 수 있을 것임을 보여주었다. 그러나 음성신호의 프랙탈 특징을

직접 사용하였을 때에는 성능의 향상이 없거나 오히려 낮아지기도 하였으므로 이에 대한 보다 많은 연구가 필요하다.

RBFN 분류기의 특성을 알아보기 위해서 학습 패턴수와 노드개수를 변화시켜 가면서 인식률을 조사하였다. 학습패턴의 수를 변화시켰을 때에는, 패턴이 많아질수록 인식률이 높아졌으나, 패턴수가 적어져서 한 화자당 115개 밖에 되지 않았을 때에도 모든 데이터를 사용했을 때에 비해 인식률이 크게 낮아지지 않았다. 이것을 통해 본 논문에서 제안한 분류기는 적은 수의 패턴으로도 전체 표본집합의 분포를 모델링하는 일반화 능력이 있음을 알 수 있다. 노드의 폭을 조절함으로써 네트워의 노드 개수를 변화시켰을 때에는 노드 개수가 어느 정도 이상일 때에 인식률이 크게 향상되었다. 또한 노드 개수가 너무 많아지더라도 인식률이 오히려 감소하는 현상은 나타나지 않았다.

4 결 론

본 논문에서는 구문독립 화자인식에서 중요한 역할을 하는 분류기를 신경회로망과 그 출력으로부터 화자를 결정하는 화자결정회로망의 조합으로 구현하였고, 특징추출과정에 있어서는 프랙탈 기하학을 응용한 특징을 사용하여 화자인식률을 높여보았다.

음성신호는 프레임 단위로 나뉘어져서 특징벡터들로 변환되며, 그것들은 각각 독립적으로 분류기에 입력된다. 특징벡터들은 프레임분류 신경회로망을 거치면서 등록되어 있는 각각의 화자에 속하는 정도가 계산되며, 그 결과는 화자결정회로망에서 MAXNET과 누적합의 과정을 통하여 최종적으로 화자를 결정하게 된다. 이러한 방법의 장점은 기존 방법이 주어진 음성 데이터를 통계적으로 분석하여 화자인식 시스템을 구축하는데 비해, 온라인 상태에서 학습에 의해 화자인식 시스템을 자동으로 구축할 수 있다는 것이다.

프레임분류 신경회로망으로는 여러가지 구조가 사용될 수 있는데, 그 중에서 변형된 RBFN을 이용한 방법은 여러가지 잇점이 있었다. 그것은 각 화자마다 개별적인 네트워을 갖게 하여 화자마다 독립적으로 학습되므로, 이미 등록된 화자의 추가적인 학습이나 새로운 화자의 등록이 아주 쉽다는 것이다. 또한, 학습 방법이 간단하고 같은 데이터의 학습을 여러번 반복할 필요가 없어서 학습이 한번에 끝난다는 장점이 있다.

RBFN 분류기의 각 화자당 파라미터의 수는 생성된 평균 노드수에 비례하며 각 노드는 중심벡터와 그 노드의 학습회수를 저장하는 변수를 가지므로, 화자당 파라미터의 수는 (평균노드수) × (특징요소개수 + 1) 이 된다. 앞에서 적당한 노드 개수가 평균적으로 67개였던 것을 기억하면 이 분류기가 많은 파라미터를 필요로 하지 않는다는 것을 알 수 있다. 그리고 본 논문에서 제시된 구조는 모두 하드웨어로 구현될 수 있는 것이므로 이것을 하드웨어로 구현할 경우 화자인식 속도를 높일 수 있다.

특징벡터에 프랙탈 차원을 사용함으로써 인식률이 크게 향상되지는 않았지만 화자에 따라서는 인식률에 많은 차이가 나타나는 것도 관찰되었으며 이 부분에 관해서는 앞으로의 연구를 통하여 그 타당성이 검증되어야 할 것이다. 앞으로 음성신호의 프랙탈 차원을 화자인식에 이용하는 것에 관한 연구는 음성을 선형시스템의 출력으로 모델링하는 것의 한계를 보완하는 측면에서 이루어져야 할 것으로 보인다.

본 논문에서 제시한 분류기의 구조는 아직 초기 단계로서 많은 부분에서 개선되어야 할 요소가 남아있다. 현재의 RBFN은 학습시에 노드를 생성하기만 하므로, 학습이 오래 진행되면 네트워크의 규모가 굉장히 커질 수가 있다. 따라서 이 점을 보완하기 위해 학습시에 잘 사용되지 않는 노드를 없애주는 것에 대한 연구가 필요하다. 또한 현재의 분류기 구조를 발전시켜서 각각의 노드에서 나온 출력을 더하거나 가중치를 부여하는 등의 방법으로 화자결정의 정확도를 높이는 것에 대한 연구가 필요하다. 이 때에는 네트워크의 구조를 최적화하는 것뿐만 아니라 보다 효율적인 학습방법을 개발하는 것에 관한 연구도 같이 진행되어야 할 것이다.

본 논문은 닫힌집합에서의 화자식별에 관한 것으로서 제안된 구조를 확장하면 등록된 화자가 아닌 사람의 데이터를 거부하는 열린집합 화자식별이나 화자검증에도 적용될 수 있을 것이다. 또한 제안된 구조에서 되먹임 구조로 되어있는 지연합을 앞먹임 구조로 바꾸어 고정된 개수의 지연과 그것의 합으로 구현한다면, 그것은 최근 몇 개의

프레임에서 가장 가능성 있는 화자를 출력으로 내는 것이 되므로 연속된 대화에서 화자의 전이를 탐지하는 용도로도 사용될 수 있다. 마지막으로 본 화자인식 시스템을 실제 환경에서 사용할 수 있기 위해서는 보다 많은 화자와 여러가지 사용환경에서 안정성을 검증받아야 한다.

참고문헌

[1] Tomoko Matsui and Sadaoki Furui. Speaker Recognition Technology. NTT Review, 7:40-48, March 1995.

[2] Tomoko Matsui and Sadaoki Furui. Concatenated phoneme models for text-variable speaker recognition. In Proc. ICASSP '93, pages II-391-II-394, 1993.

[3] George R. Doddington. Speaker Recognition Identifying People by Their Voices. Proc. IEEE, 75:1651-1664, November 1985.

[4] Bishnu S. Atal. Automatic recognition of speakers from their voices. Proc. IEEE, 64:460-475, April 1976.

[5] Aaron E. Rosenberg. Automatic speaker verification : A review. Proc. IEEE, 64:475--487, April 1976.

[6] Yu-Hung Kao, P.K. Rajasekaran, and John S. Baras. Free-text speaker identification over long distance telephone channel using hypothesized phonetic segmentation. In Proc. ICASSP '92, pages II-177-II-180, 1992.

[7] Douglas A. Reynolds and Richard C. Rose. Robust Text Independent Speaker Identification Using Gaussian Mixture Speaker Models. IEEE Trans. on Speech and Audio Processing, 3:72-83, January 1995.

[8] Frank K. Soong, Aaron E. Rosenberg, and Biing Hwang Juang. A vector quantization approach to speaker recognition. T&T Technical Journal, 66:150-162, Mar/Apr 1987.

[9] A. Higgins, L. Bahler, and J. Porter.Voice identification using nonparametric density matching. In Automatic Speech and Speaker Recognition, pages 211-232. Kluwer Academic Publishers, 1996

[10] A. L. Higgins, L. G. Bahler, and J. E. Porter. Voice identification using nearest-neighbor distance measure. In Proc. ICASSP '93, pages II-375 - II-378, 1993.

[11] Laszlo Rudasi and Stephen A. Zahorian. Text independent Talker Identification with Neural Networks. In Proc. ICASSP '91, pages 389-392, 1991.

[12] J. Oglesby and J. S. Mason.Radial basis function networks for speaker recognition. In Proc. ICASSP '91, pages 393-396, 1991.

[13] Y. Bennani and P. Gallinari. On the use of TDNN extracted features information in talker identification. In Proc. ICASSP '91, pages 385-388, 1991.

[14] Jacek M. Zurada. Introduction to Artificial Neural Systems. PWS Publishing Company, Boston, MA, 1995.

[15] J. D. Markel and A. H. Gray, Jr. Linear Prediction of Speech. Springer-Varlag, Berlin, 1980.

[16] L. R. Rabiner and R. W. Schafer. Digital Processing of Speech Signals. Prentice Hall, New Jersey, 1978.

[17] Francis C. Moon. Chaotic and Fractal Dynamics. John Wiley & Sons, New York, 1992.

[18] Jens Feder. Fractals. Plenum Press, New York, 1988.

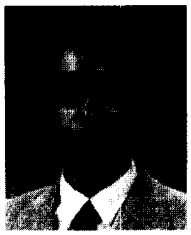
[19] P. P. Ohanian and R. C. Dubes. Performance evaluation for four classes of textural features. Pattern Recognition, pages 819-833, 1992.

[20] Jonathan M. Blackedge. On the synthesis and processing of fractal signals and images. In Applications of Fractals and Chaos. Springer-Verlag, Berlin, 1991.

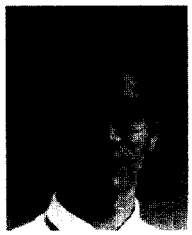
[21] Tomaso Poggio and Federico Girosi. Networks for approximation and learning. Proceedings of the IEEE, 78:1481-1497, September 1990.

[22] Simon Haykin. Neural Networks. MacMillan, 1994.

[23] Hiroaki Sakoe and Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. IEEE trans. ASSP, pages 43-49, February 1978.



이 종 은(Jong-Eun Lee)



최 진 영(Jin Young Choi)