비지도 학습 방법을 적용한 모듈화 신경망 기반의 패턴 분류기 설계*

(A Design of Classifier Using Modular Neural Networks with Unsupervised Learning)

최 종 원* (Jong-Won Choi)　　오 경 환** (Kyung-Whan Oh)

요 약 논문에서는 모듈화 신경망을 이용한 비지도 학습방법의 분류기를 제안한다. 각 모듈은 데이터의 통계학적인 분석의 결과로 설계되어져서, 데이터의 독립적인 군집들을 나타내게 된다. 이런 신경망의 독립적인 분류 결과와 근접거리 척도를 이용한 유사도 측정을 통해 더욱 정확한 분류를 가능케 하며, 오분류를 하는 모듈을 삭제함으로써 계산량을 줄인다. 이런 과정을 통해 신경망에 사용되는 각종 변수에 대한 별다른 조사 과정 없이 최상의 성능을 발휘하는 신경망에 준하는 성능을 가진 신경망을 구축했다.

연구 세부 분야 비지도 학습, 모듈화 신경망
주제어 비지도, 신경망

Abstract In this paper, we propose a classifier based on modular networks using an unsupervised learning method. The structure of each module is designed through stochastic analysis of input data and each module classifier data independently. The result of independent classification of each module and a measure of the nearest distance are integrated during the final data classification phase to allow more precise classification. Computation time is decreased by deleting modules that have been classified to be incorrect during the final classification phase. Using this method, a neural network sharing the best performance was implemented without considering, lots of of variables which can affect the performance of the neural network.

Keywords Neural Network, Unsupervised Learning

1. 서 론

패턴분류는 주어진 많은 데이터 중에서 유사한 특성을 가진 것끼리 묶는 것을 말한다. 대부분의 패턴 인식은 이런 패턴분류를 통해 가능하며, 패턴인식 방법으로는 통계를 이용한 방법과 구문론적 패턴인식 방법, 그리고 신경망을 이용한 방법 등이 있다. 일반적으로 분류되는 데이터들은 이진논리에 근거해서 분류하기는 힘들기 때문에 퍼지 집합 이론이 요구되기도 한다. 이는 데이터의 주관적인 내용을 수치적으로 해석할 수 있기 때문에 비선형적인 데이터 분류 문제 등에 많이 활용되고 있다.

패턴분류에서 사용되는 전통적인 방법으로는 패턴 집단을 분리하는 직선이나 곡선을 표현하는 결정함수를 이용하는 방법과 패턴 벡터를 유클리드 공간상의 한 점으로 간주하고 거리함수를 사용해 점간의 근접 정도를 통해 유사도를 측정하여 클래스를 결정하는 방법이 있다. 거리함수를 이용하는 방법에는 패턴 부류의 변형 정도가 제한되어 있을 때에 효율적으로 작동하는 최소 거리 패턴 분류방법과 각 군집에서 군집을 특징 지우는 모델이나 군집의 중심을 통해 최소 거리 개념을 적용하는 군집 탐색 방법이 있다. 이 방법 중에서 잘 알려진 Isodata 알고리즘은 매 반복 단계마다 표본의 평균을 군집의 중심으로 정하는 점에서 K-means 알고리즘과 비슷하지만, 일련의 부가적인 선험적 절차들을 대화 기법으로 통합하였다는 점에서 K-means 알고리즘과는 다르다.

* 본 연구는 한국학술진흥재단 '97 자유공모과제, 지원사업에 의하여 연구되었음.
* 서강대학교 컴퓨터학과 학·석사 졸업
** 서강대학교 컴퓨터학과 교수

통계적 패턴 인식 접근방법은 데이터의 정확한 분포를 얻는다는 가정 하에서, 입력 데이터로부터 추출된 특징 값을 사용해 특징 벡터들을 각각의 클래스로 배정한다. 이 방법은 임의의 패턴의 분류 규칙을 유도할 수 있고 유도된 분류 규칙은 평균 개념에 바탕을 둔 통계적인 접근 방법을 사용함으로써 오분류 확률을 줄일 수 있다는 점에서 최적이다.[5]

구문론적 패턴 인식 방법은 특징간의 상호 관련성 또는 상호 연결성 정보와 같은 구조적 정보를 이용하는 방법이다. 이 방법을 사용하려면 구조적 정보를 정량화하여 추출할 수 있어야 하며 패턴의 구조적 유사성을 평가할 수 있어야 한다. 전형적으로 이 방법은 간단한 부분 패턴으로 구성된 복잡한 패턴을 계층적으로 묘사한다.[4]

위의 방법들 외에 분류기를 작성하기 위해 새롭게 사용되는 방법이 신경망을 이용하는 방법이다. 신경망 분류기는 생물의 신경망적 계산 방식을 모델로 하고 있다. 신경망 분류기는 뉴런 또는 처리 요소에 해당하는 활성화 함수, 네트워크의 구조, 유니트간의 연결 형태와 강도를 적절하게 조정하는 학습 규칙 등에 따라 여러 가지 형태의 모델로 구분될 수 있으며, 대표적인 예로 Hopfield network, Hamming Network, Perceptron, Multilayer Perceptron, Self-Organizing Feature Map등이 있다.[1, 4]

이러한 모델들을 이용한 분류기는 사전에 분류할 데이터의 정보를 알고 있느냐의 여부에 따라 두 가지로 설계할 수 있다. 하나는 이미 알고 있는 정보를 바탕으로 분류기를 학습시키는 BPN(Back Propagation Neural Network) 같은 지도 학습 신경망이고 다른 하나는 그런 정보를 갖지 않은 상태에서 분류가 이루어지는 SOM(Self-Organized Map)과 같은 비지도 학습 신경망이다.

본 논문에서는 사전에 어떤 데이터의 정보도 주지 않은 채 분류기를 구성하는 비지도 학습 신경망을 구현하고 있다. 특히, 대부분의 비지도 학습 신경망이 분류되어질 클래스의 수를 대강 예측하는 값을 사용한다는 점에 대해 본 논문에서는 데이터의 사전 정보를 전혀 주지 않고 있다. 논문에서 구현한 신경망은 모듈화 신경망의 구조적 특징을 이용하였다. 여러 개의 모듈들이 출력하는 결과를 토대로 클래스에 대한 분류를 얻게 된다. 각 모듈의 구조는 비지도 학습 신경망이며 크기는 각기 다르다. 이 분류기에 마지막 단계에 있는 클래스 결정부에서 각 모듈의 결과를 종합하여 최종적으로 분류 결과를 출력하게 된다.

2. 패턴 분류기와 모듈화 신경망

2.1. 패턴 분류기(Pattern classifier)의 개요

패턴 분류기의 형태나 방법은 다양하지만 그들의 공통된 목적은 정확한 패턴 분류이다. 아래는 패턴 분류기를 설계할 때 고려되는 사항이다.

On-Line Adaptation : 기존의 분류 정보에 새로운 클래스를 분류해 낼 수 있도록 학습될 수 있어야 한다. 기존의 분류기는 대부분 Off-Line Adaptation이다.

Nonlinear Separability : 패턴 분류에서 가장 어렵던 문제였지만, 퍼지 분류기나 신경망을 이용한 분류기 등이 나오면서 더 이상 문제되지 않는다.

Overlapping Classes : 겹쳐진 클래스 영역을 분류하는 문제이다. 통계학적인 확률 분포를 알 수 있다면 Bayes Classifier를 구성하는 것이 가장 좋은 성능의 분류기이다.

Training Time : 대부분의 분류기는 Object Function을 최대화하거나 최소화하는 방법으로 문제를 해결하므로 시간이 많이 걸린다.

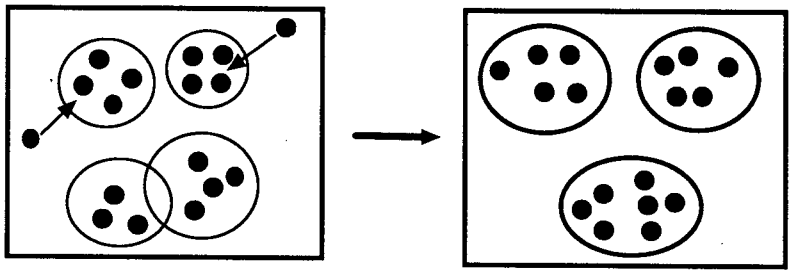
Soft and Hard Decision : 패턴 분류기는 Soft decision과 Hard decision을 할 수 있어야 한다. Hard decision은 클래스의 구분 여부를 [0,1]로 표현하는 것이고, Soft decision은 [0~ 1]사이의 값으로 표현한 것이다.

Verification and Validation : 패턴 분류 결과를 확인하고 평가할 수 있어야 한다.

Turning Parameter : 가장 이상적인 패턴 분류기에서는 조정해줄 어떠한 변수도 필요가 없다. 변수가 존재한다면 변수에 의해 어떤 식으로든지 분류기가 영향을 받는다는 것이기 때문이다.

Non-parametric : 사전 정보 없이 패턴 분류를 하는 분류기를 non-parametric분류기라고 한다.

이런 조건들이 이상적인 패턴 분류기가 고려하여야 할 사항들이다. 본 논문에서는 이런 사항들 중에서 On-Line Adaptation 지원, Nonlinear Separability의 해결, Soft and Hard Decision 가능, Tuning Parameter 불필요, Non-parametric한 구조 등으로 설계되어서 언급한 부분의 상당 부분이 만족되도록 설계하였다.



〈그림 1〉 근접 개념으로 이루어지는 군집화

2.2. 군집 탐색

분류기를 이용해서 데이터를 분류하게 되면 일반적으로 예상되는 클래스의 갯수 이상이나 이하의 분류 결과가 나오곤 한다. 예상된 클래스 갯수 이하의 분류 결과가 나오는 것은 분류가 실패한 것이지만 클래스 갯수 이상으로 분류 결과가 나오는 경우 재분류해야 한다. 군집 탐색의 방법에는 Nearest Neighbor 방법이나 C-means 알고리즘이 있다.

Nearest Neighbor(1-NN) 분류 방식은 어떤 클래스에 속해 있는지 모르는 패턴 p를 가장 가까운 거리에 있는 클래스로 분류한다. 퍼지 C-means 알고리즘은 각 데이터의 점과 각 군집 중심과의 거리를 통한 유사도 측정을 기반으로 한 목적함수의 최적화 방식을 이용한다. C-means 알고리즘은 퍼지 C-means 알고리즘의 특별한 경우이다. 이러한 퍼지 C-means 알고리즘은 최적의 퍼지분할(fuzzy partition), 패턴 분류(pattern classification)와 영상분할(image segmentation)등에도 널리 이용되고 있다. 주어진 데이터들로부터 군집을 특징짓는 모델이나 군집의 중심을 결정하는 능력은 최소 거리 개념에 기반을 둔다. [1,4,5] (그림 1)

사용될 군집화 기준(clustering criterion)은 heuristic 기법으로 표현하거나 특정한 성능 지표의 최소(또는 최대)화에 바탕을 둘 수 있다. Heuristic 기법은 경험과 직관을 바탕으로 한 방법으로, 패턴을 군집 영역에 배정하기 위해 선택한 유사도를 이용하는 규칙들로 구성되며 유클리드 거리 척도가 근접 척도로서 쉽게 해석되므로 이 접근 방식에 적합하다.

그러나 두 패턴간의 상대적인 유사도이기 때문에, 수용 가능한 유사도를 정의하기 위한 임계값을 설정할 필요가 있다. 성능 지표 접근 방법은 선택된 지표를 최대화하거나 최소화하는 절차이다. 가장 자주 사용되는 지표는 제곱 오차의 합 지표로 아래와 같다.

$$J=\sum_{j=1}^{N_c}\sum_{x\in S_j}\|x-m_j\|^2$$

$$m_j=\frac{1}{N_j}\sum_{x\in S_j}x$$

N_c : 군집 영역의 갯수, S_j : j번째 군집 영역에 속하는 표본 집합

m_j : 집합 S_j에 속한 표본의 평균 벡터

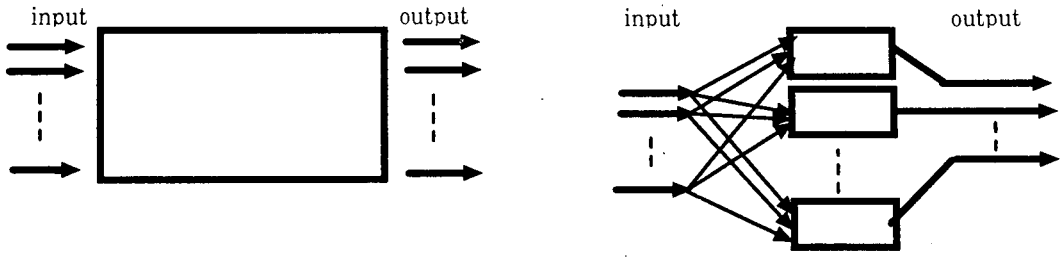
N_j : S_j에 속하는 표본의 갯수

이밖에 사용되는 지표로는 한 군집 영역에 있는 표본간의 평균 제곱 거리, 다른 군집 영역에 속하는 표본간의 평균 제곱 거리, 다른 군집 영역에 속하는 표본간의 평균 제곱 거리, 산포 행렬의 개념에 기반을 둔 지표, 최대 최소 분산 지표 등이 있다.[1,8,9]

2.3 모듈화 신경망(Modular neural network)

모듈화 신경망(Modular neural network)은 전체적인 신경망 구조가 모듈로 이루어진 신경망이다. 각 모듈의 구조는 신경망으로 이루어져 있고 모듈의 연결 방식에 따라 다른 목적에 사용되며 각각 다른 특성을 보인다. 이 구조의 장점은 거대한 신경망을 구성하기 보다는 여러 개의 모듈을 연결함으로써 전체 연결의 수를 줄일 수 있으며 모듈들의 병렬 처리를 통해 속도 향상을 꾀할 수 있다는 것이다. (그림 2 참조)

모듈화 신경망(Modular neural networks)을 이용한 분류기에서는 각각의 모듈이 하나의 클래스를 학습하게된다. 이로 인해 K-Class문제를 K개의 2-class문제로 변환시키는 것으로써 보다 정확한 분류를 할 수 있게 된다.[2,3] 일반적으로 모듈화 신경망(Modular neural networks)이 비모듈화 신경망(non-Modular neural networks)에 대해서 가지게 되는 장점은 다음과 같다.



〈non-modular neural network〉 〈modular neural network〉

〈그림 2〉 Modular architecture

1. 모듈화 신경망(Modular neural networks)은 비모듈화 신경망(non-modular neural networks)보다 적은 학습 횟수에 의해서도 비슷한 결과를 가진다.
2. 일반적으로 모듈의 크기는 비모듈화 신경망(non-modular neural networks)보다 작기 때문에 1회 학습 시간이 빠르다.
3. 모든 모듈에서 사용된 변수가 비모듈화 신경망(non-modular neural networks)보다 많음에도 불구하고 더 빠른 수렴(convergence) 속도와 더 좋은 일반화(generalization) 성능을 보인다.
4. 모듈은 독립적인 학습을 하기 때문에 병렬처리가 가능하다. 병렬처리가 가능한 시스템에서는 더 빠른 수행 시간을 보여주게 된다.

본 논문에서는 각 모듈들은 비지도 학습 신경망으로 이루어져 있으며 각 모듈들은 서로 영향을 주지 않는 독립적인 학습을 하게 된다.

2.4 비지도 학습 방법의 신경망

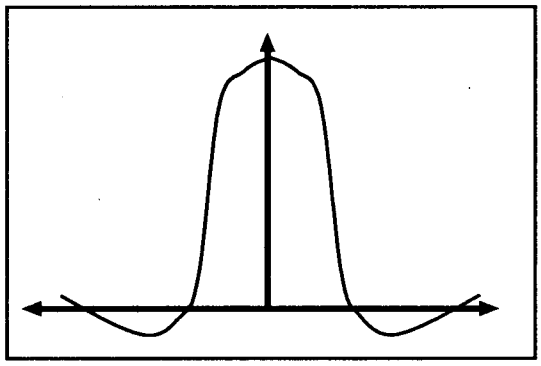
일반적으로 많이 쓰이는 비지도 학습 신경망으로는 자기 조직화 지도(Self-Organizing Map : SOM)가 있다. 생명체가 가지는 학습 능력을 수학적으로 모델링한 이 방법은 사전 정보가 없는 문제에 대해서 탁월한 성능을 보이는 신경망이다.

코호넨의 자기 조직화 네트워크는 기존의 지도 학습 방법의 신경망들에 비해 구조가 간단하다. 이 신경망은 두 개의 층으로 이루어져 있는데, 첫 번째 층은 입력층(Input Layer)이고 두 번째 층은 경쟁층(Competitive Layer)인데 2차원의 격자로 이루어져 있다. 모든 연결은 첫 번째 층에서 두 번째 층의 방향으로 되어 있으며 두 번째 층은 완전 연결(Fully Connected)되어 있다.

코호넨 네트워크(Kohonen network)라고도 불리는 이 신경망을 사용하기 위해서 두 가지 일을 해야 하는데 하나는 층 내의 뉴런 연결강도 벡터가 임의값을 가지면서 적합하게 최적화 되어야 하는 것이고 다른 하나는 연결강도 벡터와 입력벡터의 값이 통상 0에서 1사이로 정규화(normalized)되어야 하는 것이다.

이 논문에서 사용되는 신경망 역시 자기 조직화 지도와 비슷한 특성을 가진다. 학습을 원활하게 진행하기 위해서 학습 벡터들간의 차이를 일정하게 하는 정규화 과정을 거치게 된다. 하지만, 이 과정에서 비선형 정보가 사라질 수도 있고, 넓은 범위의 값을 가지는 벡터가 좁은 범위의 값으로 매핑 됨으로써 잘못된 학습을 유도하기도 한다. 정규화 방법은 학습 결과에 큰 영향을 미치는 요인 중의 하나이기 때문에 분류할 데이터에 적합하게 결정되어야 한다.[1,6,7]

비지도 학습 방법의 신경망 학습방법은 대부분 '승자독점(winner take all)'을 택하고 있다. 승자 뉴런만 출력을 낼 수 있고, 승자뉴런과 그 이웃 뉴런들만이 '측면제어(lateral inhibition)'〈그림 3〉이라고 불리는 멕시칸 모자(sombero)와 유사한 형태의 그래프



〈그림 3〉 측면 제어(lateral inhibition)

와 같이 연결 강도를 조정함으로써 학습이 이루어지는 것이다. 학습이 끝나게 되면 비지도 학습 신경망은 자연스럽게 학습 데이터의 topology적 특성을 반영하는 형태가 된다.

3. 모듈화 신경망(Modular Neural Network)을 이용한 비지도 학습 방법의 분류기 설계

본 논문에서 제안한 분류기는 크게 세 부분으로 나눌 수 있다. 첫째, 데이터에 대한 통계학적인 정보를 이용하여 모듈화 신경망의 크기와 구조를 결정하기 위해 데이터의 정보를 수집하는 선행처리부, 둘째 데이터를 정규화하고 정규화된 데이터를 이용해 모듈들을 학습시키는 학습 및 분류부, 셋째 모듈들이 출력해 내는 결과와 학습된 모듈들을 이용해 최종적인 분류를 하게되는 클래스 결정부, 이 세 부분으로 이루어진다. (<그림 4> 참조)

3.1 선행처리부

대부분의 분류 문제에서 분류할 데이터가 충분치 않은 경우가 대부분이므로 데이터를 통계적으로 분석하여 데이터 분류를 위한 사전 정보를 얻는다는 것은 대단히 어렵다.

하지만 데이터의 통계학적인 특성을 분석해서 비지도 학습 신경망의 구조를 결정할 때 참고 자료로 활용할 수 있다. 학습할 데이터들의 평균과 분산을 이용해 학습 데이터들이 어느 구간에 분포되어 있는 지를 알 수 있고, 이를 통해 학습이 잘 될만한 신경망의 크기를 예상할 수 있다. 신경망의 크기를 결정하기 위하여 optimal_size라는 함수를 정의하였다.

$$optimal\_size(\sigma_1, \sigma_2, \cdots, \sigma_n) = \left| c * \sqrt{\frac{1}{\sigma_1 * \sigma_2 * \cdots * \sigma_n}} * n \right|$$

n : 학습 벡터의 차원,

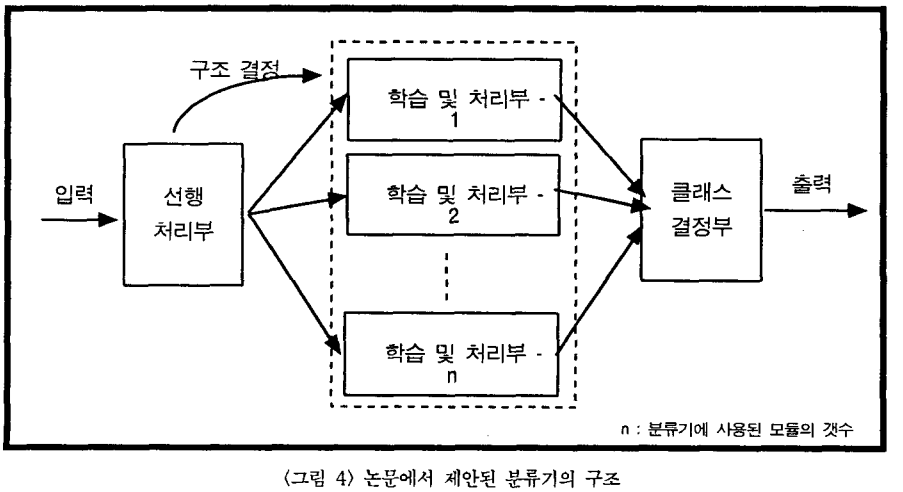
$\sigma_1, \sigma_2, \cdots, \sigma_n$: 학습 벡터 각각의 표준편차

비지도 학습 방법은 신경망의 weight공간에 학습 벡터를 대표할 수 있는 벡터를 표현하는 것이고 이들 벡터들의 표준편차가 적으면 학습 벡터를 표현하는 weight 공간을 늘려줘야 하고, 분산 정도가 크면 weight 공간을 줄여줌으로써 적절한 비지도 학습 신경망의 크기를 알 수 있게 된다. c는 optimal_size가 적절한 정수가 되도록 곱해주는 상수 값이며, 비지도 학습 신경망의 출력 노드의 갯수는 $(optimal\_size)^2$ 개다.

분류기에서 사용되는 모듈의 수가 클수록 정확도와 신뢰도가 올라가는 것은 쉽게 예상할 수 있다. 하지만 정확한 분류를 위한 모듈의 갯수에 대한 정확한 사전 정보를 얻기 힘들기 때문에 실험 대상과 실험에 사용된 데이터로부터 능력에 맞게 조정할 필요가 있다.

3.2 학습 및 분류부

비지도 학습 신경망의 경우 학습 벡터에 대한 정규화 과정은 학습 결과와 속도에 많은 차이를 일으키는 요인 중의 하나이다. 일반적으로 많이 쓰이는 비지도 학습 신경망인 SOM의 경우에는 [0.0 ~ 1.0]의 값으로 학습벡터를 정규화 한 후에 사용한다. 본 논문에서는 [-1.0 ~ 1.0]의 값으로 정규화 했다. [-1.0 ~ 1.0]의 값은 [0 ~ 1.0]의 값보다 범위가 넓기 때문에 벡터들이 비슷한 값을 갖게되는 오류의 발생을 줄



<그림 4> 논문에서 제안된 분류기의 구조

일 수 있다. 그러나 [-1.0 ~ 1.0]보다 더 큰 범위를 사용하면 벡터들의 표현 범위는 더 늘어날 수 있으나, 그로 인해 학습에 더 악영향을 끼치게 되므로 [-1.0 ~ 1.0]의 값으로 제한하였다.

정규화 과정은 일반적으로 선형변환을 하고 분류할 데이터들은 선형적, 비선형적인 관계 등을 가질 수 있다. 분류하기 힘든 데이터는 비선형 관계를 가진 데이터인데 이를 비선형 변환으로 정규화 시킬 경우 본래의 데이터 특성을 잃어버릴 수 있기 때문에 선형 변환을 하였다.

모듈화 신경망(Modular neural network)에서 사용되는 각 모듈들의 신경망의 크기는 선행처리부에서 얻어진 optimal_size값을 기준으로 (optimal_size)2, (optimal_size±1)2, (optimal_ size±2)2, …, (optimal_size±n)2 개의 출력 노드를 갖게 하였다. 이렇게 다양한 크기의 신경망을 학습시키는 이유는 신경망을 통한 데이터의 분류 결과가 더욱 다양하기 때문에 같은 크기들의 신경망에 의한 오분류를 피하고 정확한 결과를 얻기 위함이다. 각 모듈들은 비지도 학습 신경망이며 코호넨(Kohonen)의 학습 규칙을 사용하였다. weight를 조정하는 코호넨의 학습 규칙은 다음과 같다. [1, 11]

$$w_{ji}(t+1) = w_{ji}(t) + \eta(t)(x_i(t) - w_{ji}(t))$$

j : 승자뉴런 j*을 비롯한 학습에 참여한 이웃한 노드

η(t) : mexican hat sombero 형태의 이득항 (gain term)

모듈에 사용되는 비지도 학습 신경망의 알고리즘을 정리하면 다음과 같다.

〈비지도 학습 신경망의 알고리즘〉

단계 1. 연결 강도를 -1.0과 1.0 사이로 초기화한다.
N 개의 입력으로부터 M개의 출력 유니트 간의 연결강도를 작은 값의 임의의 수로 초기화한다. 〈그림 5〉에서와 같이 초기 이웃 반경은 모든 유니트가 포함될 수 있도록 충분히 크게 잡았다가 점차 줄여나간다.

단계 2. 학습벡터들을 -1.0과 1.0 사이로 정규화하여 제시한다.

단계 3. 입력과 모든 출력 노드간의 거리를 계산한다.

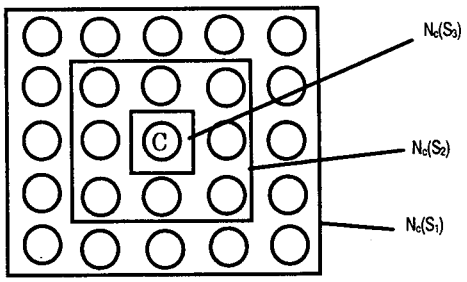


그림 5) 이웃 반경의 크기 조정

입력과 모든 출력 유니트 j간의 거리 d_j는 다음의 식과 같이 계산한다.

$$d_j = \sum_{i=0}^{N-1}(x_i(t) - w_{ji}(t))^2$$

여기서 $x_i(t)$는 시간 t에서의 i 번째 유니트에 대한 입력이고, $w_{ji}(t)$는 시간 t 에서 i 번째 입력 노드로부터 j 번째 출력 유니트간의 연결강도 이다.

단계 4. 최소 거리에 있는 출력 노드를 선택한다.
거리를 나타내는 d_j의 값 중 최소 값을 갖는 출력 유니트 j*를 선택한다.

단계 5. 출력 노드 j*와 그 이웃간의 연결 강도를 갱신한다.
유니트 j*와 $NE_{j^*}(t)$로 정의된 이웃 반경내의 모든 이웃 유니트 간의 연결강도를 다음 식과 같이 조정한다.

$$w_{ji}(t+1) = w_{ji}(t) + \eta(t)(x_i(t) - w_{ji}(t))$$

여기서, j는 j*의 이웃 반경내의 이웃이고 i는 0에서 N-1사이의 정수값이며 η (t)는 0과 1 사이의 값을 갖는 이득률로서 시간이 경과함에 따라 감소한다.

단계 6. 단계 2로 가서 반복 수행한다.

3.3 클래스 결정부

각 모듈의 결과는 약간씩 다른 결과를 가지게 된다. 여러 모듈들의 분류 결과중 보다 정확하다고 판단된

분류결과를 선택하기 위해 이들 결과들을 평가하는 방법들이 필요하다. 사전 정보가 없는 상태에서 결과를 평가하기 위해 유클리드 거리 척도로 클래스의 유사도를 측정하는 근접 척도 접근방법을 택하였다.

3.3.1 군집의 유사도 측정

학습 결과 데이터 벡터들은 몇 개의 중심값을 갖는 벡터들을 중심으로 몇 개의 군집으로 분류된다. 몇 개의 군집으로 분류된 벡터들은 같은 군집에 속하는 벡터들은 벡터들간의 거리가 가깝고, 다른 군집에 속하는 벡터끼리는 거리가 멀어지게 된다. 분류가 정확할수록 이 거리 관계는 보다 명확해진다. [8, 9]

같은 i 라는 군집에 속하는 벡터와 이들을 대표하는 중심 벡터간의 거리의 합의 평균을 dis_inter_i 라 하고, i 군집의 중심 벡터와 다른 j 라는 군집의 중심 벡터간의 거리를 dis_intra_{ij}라고 정하게 되면 이들에 의해 결정되는 값 $connectivity_{ij}$의 값은 다음과 같이 정의하였다.

$$connectivity_{ij} = \frac{dis\_intra_{ij}}{dis\_inter_i}$$

connectivity의 값은 같은 군집에 속하는 벡터들의 집중도를 반영하며 다른 군집과의 분리성도 나타내고 있다. 즉, 이 connectivity의 값이 클수록 같은 군집에 속하는 벡터들의 거리 오차가 작고 다른 군집과는 거리가 멀어져 Overlap 확률이 적어진다. 상대적으로 높은 connectivity 값을 가지는 군집은 다른 군집에 비해 좋은 분류를 하고 있음을 뜻하며, 모듈 내의 군집들이 모두 connectivity값이 높으면 그 모듈은 다른 모듈에 비해 정확한 분류를 하고 있다고 평가될 수 있다.

3.3.2 모듈에서의 군집 결합

모듈들은 비지도 학습 신경망으로 구성되었기 때문에 각각의 모듈은 다른 결과를 보이게 된다. 같은 데이터에 대해서도 다른 갯수의 클래스로 분류하는가 하면 서로 다른 클래스로 분류할 수도 있다. 모듈들이 분류한 클래스는 예상했던 클래스 갯수 이상 혹은 이하로 분류된다. 기대되는 클래스 미만으로 분류하는 모듈은 정확한 분류를 기대할 수 없기 때문에 제거한다.

제거되지 않은 모듈에 대해서 데이터를 N 개의 클래스로 분류하기 위해 각 군집들이 속하는 클래스를 알아야한다. 즉, 각 모듈에서 N≤K 개의 군집들을 N 개의 클래스로 묶어주는 과정이 필요하다. 이 과정은 군집과의 유사도의 정도를 통해 이루어진다. 유사한 관계를 가진 두 군집간의 connectivity값은 다른 군집들간의 값보다 상대적으로 작은 값을 가지게 되며 이런 특성을 이용해서 N≤K 개로 분류된 군집들을 N 개의 클래스로 분류시켜주게 된다.

3.3.3 모듈들의 분류 결과 결합

위에서 언급한 바와 같이 각 클래스의 connectivity 값은 모듈내에서는 물론 다른 모듈 내에서 분류된 결과와도 상대적인 정확도를 비교할 수 있으므로 connectivity 값을 데이터의 클래스에 속하는 정도의 척도로 사용할 수 있다.

모듈내에서 가장 높은 connectivity 값을 갖는 클래스는 같은 모듈내의 다른 클래스보다 정확한 분류가 이루어졌음을 의미한다. 서로 다른 모듈 내의 connectivity의 비교는 같은 범위의 weight space 상에서의 비교이기 때문에 상대적인 평가가 가능하다. 즉, 서로 다른 모듈이 같은 데이터에 대해서 같은 클래스라는 분류 결과를 내었을 때, 더 높은 connectivity의 값을 가지는 모듈이 더 정확한 분류를 했다고 볼 수 있다. 그러나, 각각의 모듈이 한 데이터에 대해 서로 다른 분류를 보였다면, 관점을 달리할 필요가 있다. 한 쪽의 모듈이 더 높은 connectivity의 값을 가진다고 결과적으로 정확하다고 할 수 없다.

모듈화 신경망(Modular neural networks)에서는 여러 개의 모듈이 동시에 같은 데이터에 대해 분류를 하기 때문에 하나 이상의 모듈이 잘못된 분류 결과를 보이더라도 나머지 모듈들에 의해 분류결과가 보정될 수 있다는 장점이 있다.

앞에서 기술된 클래스 결정부의 구체적인 알고리즘은 다음과 같이 정리된다.

〈클래스 결정부의 알고리즘〉

단계 1. P 개의 모듈 중 학습 데이터를 N개 이상의 군집으로 분류하는 S 개의 모듈을 선택한다. 선택되지 않는 (P-S)개의 모듈은 삭제한다.

단계 2. S 개의 각 모듈에 대해서 최종 군집 갯수가 N개 될 때까지 클러스터 i, j의 connectivity 값인 C_{ij} 값이 작은 클래스터 i, j를 같은

클래스터로 합해 나간다.

단계 3. S 개의 각 모듈 내에서 N개의 클래스들에 대해 같은 클래스에 속하는 군집의 connectivity 값들을 평균함으로써 최종 클래스 갯수 N개에 대한 connectivity 값을 구한다. C_i는 최종 클래스 i 에 속하는 모든 군집들간의 connectivity 값이다.

$$C_i = \sum_{j=1}^{n} \frac{C_j}{n}$$

단계 2에 의해 $C_j \in C_i$을 만족하는 class j에 대해서 계산되며, n은 최종 class i에 속하게 된 군집들의 갯수이다.

단계 4. S개의 모듈 중 각 모듈의 최종 분류 결과가 정해진 예측 클래스 갯수보다 작으면 모듈을 삭제한다. 모듈의 갯수는 S개에서 S'로 줄어들게 된다.

단계 5. 분류될 데이터들은 각각의 모듈들에서 데이터가 속하는 클래스의 connectivity값을 각각의 클래스끼리 합하여 가장 높은 값을 갖게 되는 클래스로 데이터를 분류해 낸다.

$$Class_k = \sum_{i=1}^{S} C_i$$

분류할 데이터 각 모듈에서 x가 $x \in C_j$라면 모듈 S'개에 대해 C_j의 connectivity값을 더하여 가장 높은 값을 갖는 $Class_k$로 데이터는 분류 결과가 결정된다.

4. 실험 내용 및 실험 결과

4.1 Iris data 분류 실험 방법

패턴 분류에서 가장 잘 이용되는 분류 데이터 중 붓꽃(Iris)의 데이터에 대한 분류 실험을 하였다. 150개에 대한 붓꽃에 대해 4차원 특징 벡터를 이용하여 3개의 클래스로 분류하는 실험을 하였다. 비지도 학습에서 붓꽃이 분류될 클래스의 갯수는 3개로 주어지고 논문에 서술된 바와 같이 행하였다.[9]

실험에 사용된 데이터의 갯수가 150개밖에 되지 않기 때문에 평가는 leaving-one-out 방법을 사용하였다. leaving-one-out 방법은 데이터 중 한 개의 데이터를 제외시키고 학습시킨 다음 제외시켰던 나머지 한 개의 데이터에 대한 분류 경과를 테스트함으로써 분류기의 성능을 평가하며, 한정된 데이터일 경우 쉽게 적용할 수 있다.

붓꽃 데이터는 다음의 4개의 특징 벡터로 이루어져 있다.

① 꽃받침(sepal)의 길이
② 꽃받침(sepal)의 너비
③ 꽃잎(petal)의 길이
④ 꽃잎(petal)의 너비

4 가지 특징 벡터들은 cm 단위로 측정되어 있으며, 특징 벡터들로 분류되는 붓꽃들은 Iris Setosa, Iris Versicolour, Iris Virginica의 3종류이다. 150개의 특징 벡터들의 통계학적 특징은 〈표 1〉과 같다.

150개의 데이터들은 한 클래스 당 50개씩 학습 데이터를 가지고 있다. 구체적인 실험 조건은 〈표 2〉와 같다.

4.2 실험 결과 및 분석

분류할 특징 벡터들은 서로 높은 상관관계일수도 있고 그렇지 않을 수도 있다. 서로의 상관관계가 높다면 데이터의 특징 벡터 차원을 줄일 수도 있을 것이다. 〈그림 6〉에서 보이는 바와 같이 꽃받침은 길이와 너비의 상관관계가 없는 것으로 보인다. 그러나 〈그림 7〉에서는 꽃잎의 길이와 너비는 서로 비례 관계를 가진 것으로 보인다.

데이터들의 분포에서 보듯이 3 종류의 붓꽃에 대해 3개의 군집들 중 하나는 독립적이나, 다른 두 개는 겹쳐지는 형태를 보인다. 이 겹쳐지는 두 개의 군집들을 오차를 최소화하는 방향으로 분류하는 것이 실험의 목적이다. 본 논문에서 제안된 분류기에 의한 Iris 데이터의 분류 결과는 비교적 낮은 오차값을 보여주고 있다. 〈표 3〉에서와 같이 분류 실험 결과 오류값의 범위는 약 [0.05333 ~ 0.253333]까지이다. 실제 실험 과정 중 모든 모듈에 대한 에러값의 범위는 [0.053333 ~ 1.0]이였다. 에러값이 1.0인 경우는 모듈이 예정된 클래스 갯수 분류에 실패한 경우이다. 그리고 초기 weight에 따라 학습 결과가 차이날 수도 있다.

〈표 4〉와 같이 각 모듈의 분류 성능은 다르게 나온

〈표 1〉 Summary statistic

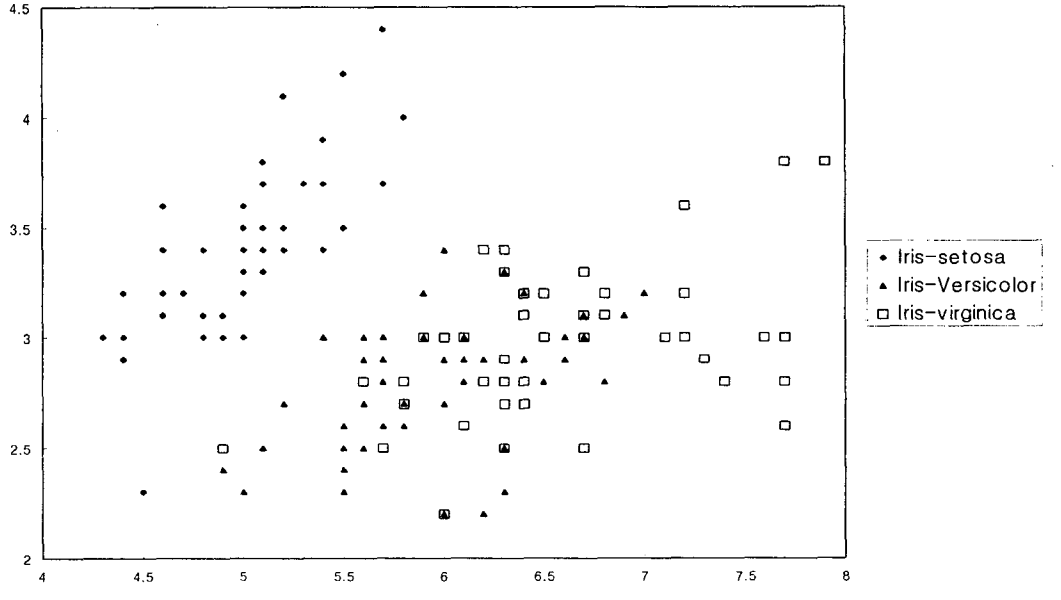
Feature Vector	Min	Max	Mean	SD	Class Correlation
Sepal length	4.3	7.9	5.84	0.83	0.7826
Sepal width	2.0	4.4	3.05	0.43	-0.4194
Petal length	1.0	6.9	3.76	1.76	0.9490 (high)
Petal width	0.1	2.5	1.20	0.76	0.9565 (high)

〈표 2〉 구체적인 실험 변수

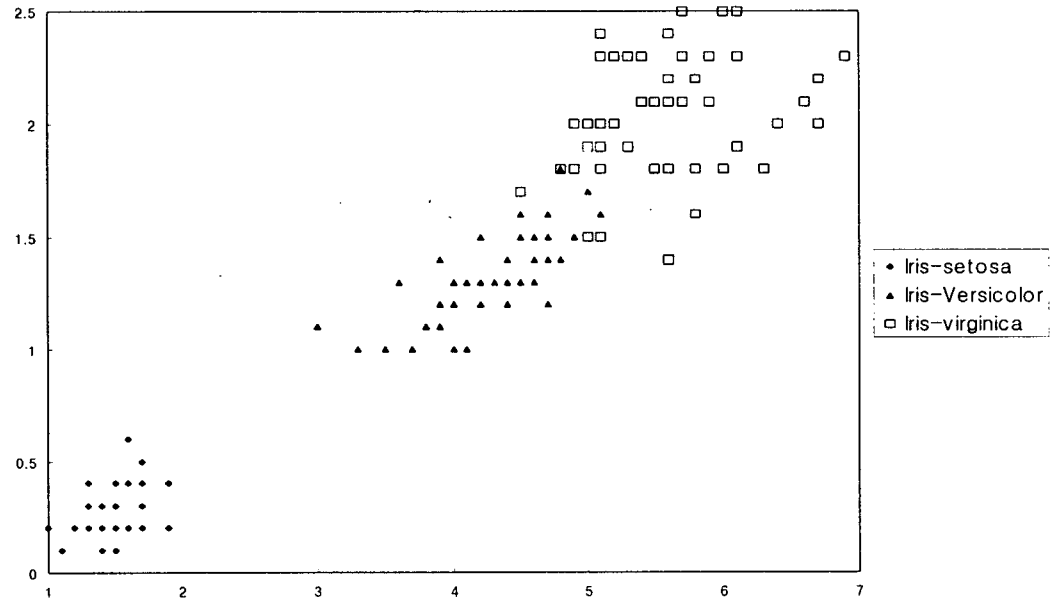
초기 학습률	학습횟수	Weight의 초기화 범위	optimal_size	Module의 갯수
0.6	10000	-1.0 ~ 1.0	5	5

다. 모듈의 결과를 종합하면 가장 분류가 잘된 모듈의 결과보다 같거나 약간 떨어졌다. 오분류한 모듈에 의한 오차 때문이다. 하지만 평균적인 에러값이 0.116933으로 에러값이 평균 이하인 경우가 56%이고 에러값이 0.15 이하인 경우가 95%인 안정적인 성능을 보여주고 있다. 이는 논문에서 제시한 실험에 대해 최고의 성능이 나오도록 변수를 조절해서 만든 코호넨 네트워크로 150회 실험을 통해 평균 Error값으로 구해진 0.1016888에 비해 모자람이 없는 성능을 보여주고 있다. 이는 제시하는 신경망 모델은 각종 변수 값을 조절하는 과정을 통해 얻어지는 최적화 되어진 코호넨 네트워크의 성능과 비슷한 성능의 신경망을 이런 과정 없이 구성할 수 있음을 보여주고 있다.

〈표 5〉는 Iris 데이터에 대해 K-means 방법을 적용한 125,000 번의 실험 중 가장 낮은 오차값을 보이는 실험 결과이다.[7] K-means방법은 데이터간의 거리를 기준으로 군집을 결정하는 방법인데 바른 처리와 데이터의 사전 정보 없이도 활용할 수 있다는 장점이 있다. 초기에 임의로 설정되는 군집의 중심에 따라 약간 다른 결과가 나올 수도 있지만 대부분 비



〈그림 6〉 꽃받침(sepal) 길이와 너비와의 분포도



〈그림 7〉 꽃잎(Petal)길이와 분포도

〈표 3〉 150번 반복 실험에 대한 Error value

Iter	Error	Iter	Error	Iter	Error
0	0.120000000	25	0.106667000	50	0.086667000
1	0.106667000	26	0.120000000	51	0.113333000
2	0.106667000	27	0.113333000	52	0.100000000
3	0.106667000	28	0.106667000	53	0.073333000
4	0.100000000	29	0.106667000	54	0.133333000
5	0.106667000	30	0.113333000	55	0.066667000
6	0.120000000	31	0.106667000	56	0.106667000
7	0.140000000	32	0.126667000	57	0.120000000
8	0.066667000	33	0.093333000	58	0.066667000
9	0.133333000	34	0.113333000	59	0.133333000
10	0.106667000	35	0.106667000	60	0.053333000
11	0.113333000	36	0.106667000	61	0.113333000
12	0.113333000	37	0.113333000	62	0.080000000
13	0.093333000	38	0.113333000	63	0.133333000
14	0.140000000	39	0.106667000	64	0.146667000
15	0.140000000	40	0.106667000	65	0.106667000
16	0.253333000	41	0.120000000	66	0.193333000
17	0.120000000	42	0.113333000	67	0.120000000
18	0.120000000	43	0.113333000	68	0.193333000
19	0.133333000	44	0.126667000	69	0.100000000
20	0.106667000	45	0.133333000	70	0.100000000
21	0.113333000	46	0.106667000	71	0.133333000
22	0.133333000	47	0.113333000	72	0.146667000
23	0.080000000	48	0.106667000	73	0.106667000
24	0.113333000	49	0.080000000	74	0.120000000

Iter	Error	Iter	Error	Iter	Error
75	0.126667000	100	0.086667000	125	0.153333000
76	0.140000000	101	0.106667000	126	0.146667000
77	0.120000000	102	0.133333000	127	0.146667000
78	0.106667000	103	0.140000000	128	0.113333000
79	0.093333000	104	0.100000000	129	0.106667000
80	0.126667000	105	0.126667000	130	0.133333000
81	0.086667000	106	0.120000000	131	0.086667000
82	0.093333000	107	0.086667000	132	0.120000000
83	0.113333000	108	0.120000000	133	0.120000000
84	0.093333000	109	0.133333000	134	0.133333000
85	0.133333000	110	0.113333000	135	0.113333000
86	0.106667000	111	0.146667000	136	0.133333000
87	0.093333000	112	0.106667000	137	0.166667000
88	0.106667000	113	0.126667000	138	0.140000000
89	0.153333000	114	0.146667000	139	0.140000000
90	0.133333000	115	0.126667000	140	0.120000000
91	0.120000000	116	0.160000000	141	0.066667000
92	0.113333000	117	0.106667000	142	0.106667000
93	0.100000000	118	0.053333000	143	0.106667000
94	0.206667000	119	0.106667000	144	0.126667000
95	0.113333000	120	0.106667000	145	0.106667000
96	0.066667000	121	0.113333000	146	0.133333000
97	0.140000000	122	0.133333000	147	0.113333000
98	0.126667000	123	0.140000000	148	0.113333000
99	0.133333000	124	0.133333000	149	0.106667000

슷한 결과를 보였다.

실험 결과를 통한 K-means방법은 비교적 오차가 큰 분류를 했다. 특히, 중복되는 2 개의 군집의 분류에서 오분류가 많이 나왔다. K-means 방법은 신경망을 이용한 방법보다 빠르고 성능 지표 J의 수렴성에 따라 항상 동일한 결과를 얻을 수 있었지만, 중복된 분류 결과의 오차값은 성능 지표 J의 특성상 아주 높았다.

비지도 학습 신경망은 초기 weight에 따라 학습 결과의 기복이 심하다. 그래서 학습이 잘 이루어지려면 초기 weight 값을 잘 조정하거나 많은 변수들을 조절해 주는 방법들이 있다. 본 논문에서 제안한 분류기는 weight에 대한 조정이나 다른 변수들의 특별한 조정 없이도 비교적 안정적인 성능을 보여주고 있다.

대부분의 비지도 학습 방법을 이용한 분류 방법들은 안정적인 분류 결과가 나오는 것을 기대하기 힘들다. 그러나 여러 모듈들의 정보들을 결합하면 한 모듈의 잘못된 결과가 다른 모듈들에 의해 보정받을 수 있기 때문에 최종 결과는 안정적일 수 있고 보다 신뢰할 수 있는 분류 결과를 얻게 된다.

이상의 결과를 토대로 본 논문에서 제안한 모듈화 신경망을 이용한 비지도 학습 방법의 분류기가 비교적 안정적으로 만족할 분류를 보임을 알 수 있다.

〈표 5〉 K-means 방법의 적용 결과

초기쌍	성능지표	최소 오차값
(0, 51, 101)	1.768368	0.24

5. 결론

본 논문에서는 모듈화 신경망의 구조를 이용한 비지도 학습 방법의 분류기를 제안했다. 일반적으로 비지도 학습 방법에서 가질 수 있는 신뢰도의 문제를 모듈화 신경망의 특성을 이용해 각각의 모듈이 독립적인 분류를 하게 함으로써 하나의 데이터를 여러번 분류한 것 같은 결과와 그 이상의 효과를 얻을 수 있다.

본 분류기에서는 각 모듈에서 얻게 되는 군집들의 connectivity 값을 결합 비교함으로써 단순히 실험의 반복이 아니라 모듈의 갯수 증가에 따라 분류의 정확도와 신뢰도를 향상시킬 수 있도록 하였다.

〈표 4 〉실험 데이터의 일부(M1,.....,M4는 분류기의 각 모듈)

Num	M0 Error	M1 Error	M2 Error	M3 Error	M4 Error	Final Error
0	1	0.106667	1	0.113333	0.253333	0.120000
1	1	0.106667	0.126667	0.113333	0.253333	0.106667
2	1	0.106667	0.126667	0.113333	0.253333	0.106667
3	1	0.106667	0.126667	0.113333	0.240000	0.106667
4	1	0.106667	1	0.093333	0.153333	0.100000
5	1	0.106667	1	0.113333	0.106667	0.106667
6	1	0.106667	1	0.113333	0.240000	0.120000
7	1	0.106667	1	1	0.140000	0.140000
8	1	0.106667	0.060000	1	0.066667	0.066667
9	1	0.106667	0.126667	1	0.133333	0.133333
10	1	0.106667	1	1	1	0.106667
11	1	0.106667	0.126667	0.113333	0.113333	0.113333
12	1	0.106667	1	0.113333	0.133333	0.113333
13	1	0.106667	0.060000	0.106667	0.066667	0.093333
14	1	0.106667	0.126667	1	0.126667	0.140000
15	1	0.106667	1	1	0.240000	0.140000

비지도 학습 방식은 지도 학습 방식에 비해 정확도가 낮지만 사전 정보가 없이 적절한 분류를 할 수 있다는 장점을 가진다. 이런 장점을 살리기 위해서는 알려지지 않은 변수의 정보를 데이터로부터 최대한 많이 얻어내야 한다.

본 논문에서는 여러 개의 모듈을 사용함으로써 데이터 부족에 의한 오분류 발생의 단점을 극복하려 하였고, 만족할만한 결과를 보여주었다. 그러나 아직 모듈의 갯수나 신경망의 초기화 등의 실험적 변수 결정이 heuristic한 방법으로 이루어진다. 때문에 이런 부분에 대해 학문적인 해석이 이루어져야 할 필요가 있고, 더 나아가서는 이들의 숫자를 줄이는 연구가 필요하다.

참고 문헌

[1] James A. Freeman / David M. Skapura, *Neural Networks*, pp 263-290, 1992

[2] Rangachari Anand, Kishan Mehrotra, Chilukuri K. Mohan, and Sanjay Ranka, Member. "Efficient Classification for Multiclass Problems Using Modular Neural Networks", IEEE Trans. on Neural Networks, Vol. 6, No 1, pp 117-124. 1995

[3] Takumi Ichimura, Eiichiro Tazaki, "Synthesis of Reflective Neural Networks eith Adaptive Module Structure", IEEE ICNN Plenary, Panel and Special Session, pp 106-111, 1996

[4] 이성환 편저, 패턴인식의 원리 I II권, 홍릉과학출판사, 1994

[5] 김대수, 신경망 이론과 응용 (I)(II), 하이테크정보, 1992

[6] Jianchang Mao, Anil K. Jain, "A Self-Organizing Network for Hyperellipsodal Clustering(HEC)", IEEE Trans. on Neural Networks, Vol. 7, No. 1, pp 16-29, Jan. 1996

[7] T. Kohonen, *Self-Organizing Maps*, Springer, 1995

[8] Patrick K. Simpson, "Fuzzy Min-Max Neural Networks - part 1 : Classification", IEEE Trans. on Neural Networks, Vol. 3, No. 5, pp 776-786, Sep. 1992.

[9] Patrick K. Simpson, "Fuzzy Min-Max Neural Networks - part 2 : Clustring", IEEE Trans. on Neural Networks, Vol. 1, No. 1, pp 32-45, Sep. 1993.