자기조직화 신경망의 정렬된 연결강도를 이용한 클러스터링 알고리즘

†이종섭* · 강맹규**

A Clustering Algorithm Using the Ordered Weight of Self-Organizing Feature Maps

Jong-Sup Lee* · Maing-Kyu Kang**

■ Abstract ■

Clustering is to group similar objects into clusters. Until now there are a lot of approaches using Self-Organizing Feature Maps (SOFMs). But they have problems with a small output-layer nodes and initial weight. For example, one of them is a one-dimension map of c output-layer nodes, if they want to make c clusters. This approach has problems to classify elaboratively. This paper suggests one-dimensional output-layer nodes in SOFMs. The number of output-layer nodes is more than those of clusters intended to find and the order of output-layer nodes is ascending in the sum of the output-layer node's weight. We can find input data in SOFMs output node and classify input data in output nodes using Euclidean distance. The proposed algorithm was tested on well-known IRIS data and TSPLIB. The results of this computational study demonstrate the superiority of the proposed algorithm.

Keyword : Clustering, Self-Organizing Feature Maps, Euclidean Distance, IRIS, TSPLIB

논문접수일 : 2005년 11월 29일 논문게재확정일 : 2006년 8월 7일
* 울산광역시중소기업종합지원센터 S/W지원팀
** 한양대학교 정보경영공학과
† 교신저자

1. 서 론

인간들은 직관적이거나 경험적으로 사물의 여러 가지 속성을 이용하여 사물을 분류하는 능력을 가지고 있었다. 예를 들어, 산에서 신기한 식물을 발견하면 먹을 수 있는지, 독이 있는지, 위험한 것인지를 판단할 능력을 가지고 있다. 이와 같은 고유한 능력을 수리적으로 체계화하고자 하는 과정을 클러스터링(Clustering) 이라고 한다. 클러스터링은 데이터(Data)를 몇 개의 클러스터(Cluster)로 대응시키는 과정이다[1, 6, 7]. 클러스터링 문제는 고차원을 갖는 n개의 데이터들을 입력으로 하여 유사한 특성을 갖는 c개의 클러스터로 나누는 것을 말한다. 한 클러스터 내의 데이터들은 높은 유사성을 보이지만, 다른 클러스터내의 데이터들과는 아주 다르다. 데이터들 사이의 유사한 정도를 평가하는 척도로서 유사도(Similarity)는 데이터의 속성 값(Attribute Value)에 기초한 유클리드 거리(Euclidean Distance)에 의하여 계산되는데 이 유클리드 거리가 작으면 작을수록 유사도가 높다는 것을 의미한다. 클러스터링의 적용분야는 고객세분화(Customer Segmentation), 패턴인식(Pattern Recognition) 등 다양하나 특히, 데이터마이닝, 마케팅, 그리고 생물학 등의 분야에 유용하다[3, 4]. 클러스터링은 데이터의 분포에 대한 이해를 구하거나 각 클러스터의 특성을 관찰하는 단독적인 도구로 사용되기도 하고, 다른 알고리즘의 전 처리 단계로 사용되기도 한다.

클러스터링 알고리즘의 종류는 크게 두 가지로 나누어지는데, 그 하나는 계층 알고리즘(Hierarchical Algorithms)이고, 나머지는 분할 알고리즘(Partition Algorithms)이다. 최근에는 신경망, 퍼지-신경망 등을 이용한 클러스터링 알고리즘이 활발히 연구되고 있다.

Mangiameli et al.[13]에 의하면 계층 알고리즘에는 두 클러스터간의 유사성을 최단거리로 측정하는 단일접합 클러스터링(Single Linkage Clustering)과 두 클러스터간이 최장거리로 유사성을 측정하는 완전접합 클러스터링(Complete Linkage Clustering), 두 클러스터간의 평균거리로 유사성을 측정하는 평균접합 클러스터링(Group-Average Clustering), 두 클러스터간의 밀도로서 유사성을 측정하는 Ward's 계층적 클러스터링(Ward's Hierarchical Clustering) 등이 있다. 계층 알고리즘은 초기에 이루어진 부적절한 병합으로 인한 문제점을 보완할 수 없는 단점을 가지고 있다.

분할 알고리즘은 데이터들을 분할하여 같은 클러스터 내에 있는 데이터들 사이의 유사한 정도는 다른 클러스터에 있는 데이터들 보다 유사한 정도가 크도록 클러스터를 형성한다. 분할 알고리즘에는 대표적으로 c-Means 알고리즘과 ISODATA 알고리즘이 있다. c-Means 알고리즘은 각 데이터와 각 클러스터의 중심 값과의 거리 차이를 최소화시키는 방향으로 알고리즘이 반복됨으로서 각 클러스터에 대한 데이터들이 재배열이 가능하도록 한 알고리즘이다. 이것은 계층 알고리즘에서 초기에 이루어진 부적절한 병합으로 인한 문제점을 보완할 수 없는 단점을 극복하였다. ISODATA 알고리즘은 c-Means 알고리즘과 같이 c개의 중심을 가지고 시작하지만 반드시 클러스터의 개수가 c개가 되는 것은 아니다. 이것으로 인하여 더욱 유동적으로 알고리즘 수행 중에 클러스터의 개수를 변할 수 있다. ISODATA 알고리즘은 고정적인 클러스터 개수인 c-Means 알고리즘의 단점을 어느 정도 보완하였다.

대부분의 클러스터링 알고리즘은 다차원 데이터를 클러스터링 하는데 좋은 결과를 제공하지 못하고 있다. 이것은 데이터 자체의 고유한 희박성이 원인이 되고 있다. 이와 같은 문제를 해결하기 위해 우선 특징 추출(Feature Selection)을 먼저 시행하여 그 차원을 줄여서 저차원 공간을 갖는 데이터를 이용하여 클러스터링 하는 알고리즘을 적용하기도 한다. 데이터 상에 존재하는 특정 차원만을 선택하고, 나머지 차원들은 잡음으로 간주하여 제거하는 알고리즘은 특정 차원만을 미리 추출하여 사용하게 됨으로써 정보의 손실을 가져올 수 있다.

신경망을 이용한 클러스터링 알고리즘은 Kohonen [10-12], Carpenter와 Grossberg 등에 의하여 제안

되었다. Kohonen은 자기조직화 신경망과 LVQ 알고리즘을 제안하였는데, 이 알고리즘들은 데이터와 클러스터의 중심 값과의 거리를 최소화 시키는 학습 알고리즘에 따라 클러스터링 한다는 점에서 c-Means 알고리즘과 대응된다고 할 수 있다.

클러스터에 대한 정보를 미리 알고 학습과정에서 정보를 반영할 수 있는 감독학습과 달리 자기조직화 신경망은 무감독학습을 사용하는 신경망의 일종으로 출력노드에 입력데이터의 유사성을 반영하는 특성지도(Feature Maps)를 스스로 형성한다. 즉, 자기조직화 신경망은 각 각의 입력데이터를 출력노드로 대응할 수 있는 능력을 가진다.

Huntsberger와 Ajjimarangsee[8]는 학습률과 이웃범위 등의 파라미터를 변경하면서 Kohonen의 학습방법을 변형한 클러스터링 알고리즘을 제시하였다.

Pal et al.[14]는 초기 연결강도에 의해 입력데이터와 출력노드간의 거리에 대한 가중 값을 준 손실함수를 정의하고, 이 손실함수를 최소화 시키는 경쟁학습 신경망 알고리즘을 제시하였다.

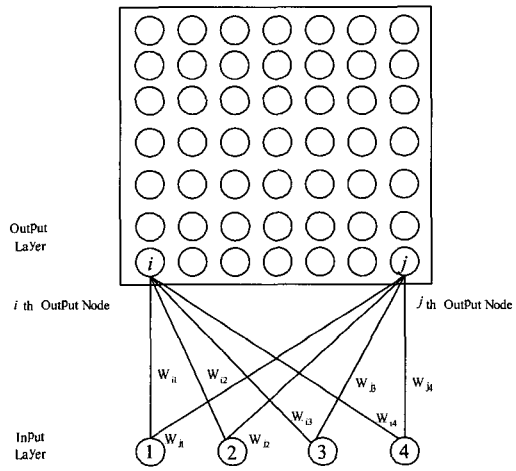
Tasao et al.[16]과 Karayiannis[9]은 신경망에 퍼지(Fuzzy)개념을 결합한 방법을 제시하였다. 특히, Karayiannis은 FALVQ(Fuzzy Algorithm for Learning Vector Quantization)입력데이터와 LVQ(Learning Vector Quantization)의 연결강도 사이의 제곱 유클리디언 거리의 가중 합을 최소로 하는 알고리즘을 개발하였다.

클러스터링 문제는 고전적인 최적화 문제이며, NP-Complete 문제로 알려져 있다. 이것은 그룹을 형성해야 할 기계의 개수가 많은 경우에는 계산량이 지수적으로 증가하여 많은 시간이 소요된다는 것을 의미한다. 따라서 최적화 해법보다 발견적 해법(heuristic algorithm)을 많이 사용한다.

본 연구에서는 Anderson의 IRIS 데이터[2]와 TSPLIB 데이터[15]를 입력데이터로 하여 자기조직화 신경망이 무감독학습을 진행됨에 따라 입력데이터와 유사한 형태로 변형되는 연결강도의 특성을 이용한다. 본 연구에서 결정해야 할 매개변수는 1차원 출력노드의 개수, 학습률, 이웃의 범위 등이다. 제안하는 알고리즘은 이와 같은 매개변수를 이용하여 학습을 진행하고, 출력노드 u와 v사이의 연결강도 거리(Weight Distance) $WD(u,v)$의 크기에 따라 그룹을 형성하였다. 그 결과 다른 클러스터링 알고리즘과 비교하여 오 분류의 개수(이동거리)를 최소화하는 알고리즘을 제시한다.

2. 자기조직화 신경망의 일반적인 고찰

SOFMs은 Kohonen[10-12]에 의해 제시된 경쟁학습 신경망(Competitive Learning Neural Network) 모델이다. SOFMs의 구조는 [그림 1]과 같이 m개의 입력노드로 이루어진 입력층과 n(즉 $p\times p$)개의 출력노드로 이루어진 출력층 두 개의 층(Layer)으로 구성되어 있다.



[그림 1] SOFMs의 일반적인 구조

입력층은 입력데이터를 받아 이들을 출력층에 사상(mapping)한다. 출력층은 1차원 또는 2차원 구조를 사용한다. 출력노드의 개수는 입력데이터가 서로 다른 출력노드에 대응될 수 있도록 입력데이터의 개수보다 큰 수로 하거나 사용자가 원하는 임의의 개수 c로 정할 수 있다. 출력층의 각 노드에는 입력데이터가 사상된다.

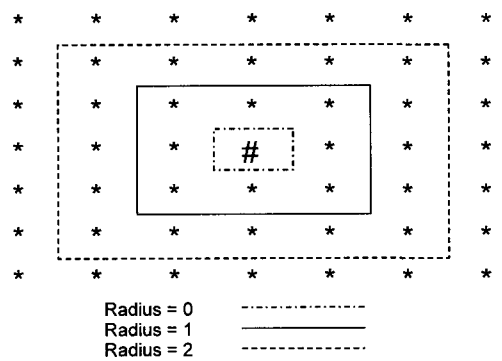
입력층의 모든 노드와 출력층의 모든 노드는 연결되어 있고, 출력노드 $i(1 \le i \le n)$와 입력노드 $k(1 \le k \le m)$사이의 연결선은 연결강도(Connecting Weight) ω_{ik}를 가진다. 연결강도는 0과 1사이의 실수로서, 초기에는 임의로 주어지지만 입력데이터에 따라 조절된다. 각 입력데이터에 대하여 이와 가장 유사한 출력노드인 승자노드 i^*를 결정하는데, 이는 식 (1)과 같이 입력데이터 $X(x_1, x_2, \cdots, x_m)$와 연결강도 ω_i 사이의 거리 D_i를 계산하고, 이중에서 가장 작은 출력노드 i^*로 정한다.

$$D_i = |x_1 - w_{i1}| + |x_2 - w_{i2}| + \ldots + |x_m - w_{im}| \quad i = 1, 2, \cdots, n \tag{1}$$

출력노드 i^*의 연결강도(W_{i*})는 식 (2)와 같이 n개의 출력노드 중에서 입력데이터 X와 가장 가까운 연결강도이다.

$$|X - W_{i*}| = \min |X - W_i| \quad \forall_i \tag{2}$$

승자노드 i^*의 앞과 뒤에 위치한 노드를 이웃노드(Neighbor Node)이라고 하는데, 이웃범위(Neighborhood) $N_{i*}(\delta)$는 승자노드 i^*로부터 δ만큼 떨어진 이웃노드의 집합이다. [그림 2]는 승자노드를 "#"으로 표시하고, 그 외의 출력노드를 "*"로 표시할 때 이차원 출력층에서 사각그리드(Rectangular Grid)를 사용하여 반경(Radius)이 $\delta = 0$, 1, 2인 경우로 이웃노드를 표현하였다.



[그림 2] 이차원 출력층에서 사각 그리드를 이용한 이웃범위

각 출력노드 i의 연결강도는 식 (3)과 같이 이웃범위에 속하느냐($i \in N_{i*}(\delta)$) 그렇지 않느냐($i \notin N_{i*}(\delta)$)에 따라 다르게 적용된다.

$$\frac{dW_i}{dt} = \alpha(t)(X - W_i) \quad i \in N_{i*}(\delta)$$
$$\frac{dW_i}{dt} = 0 \quad i \in N_{i*}(\delta) \tag{3}$$

$N_{i*}(\delta)$를 이웃범위(neighborhood)이라고 하면 자기조직화 신경망의 연결강도는 각 입력데이터에 대하여 이웃범위와 학습률을 감소시키면서 이웃범위가 승자노드 자신이 될 때까지 승자노드와 그 이웃노드의 연결강도를 식 (4)와 같이 조절한다. 학습함수 $\alpha(t)$는 시간 t의 흐름에 따라 입력데이터와 기존의 연결강도간의 차이를 조정하는 비율이다. 이 값은 0과 1사이의 실수이고, 학습이 진행함에 따라 점차 줄여나간다. 일반적으로 학습률이 너무 크면 학습이 제대로 되지 않고, 너무 작으면 학습시간이 오래 걸린다[10].

$$W(\neq w)_{ik} = W(old)_{ik} + \alpha(t)(X_i - W(old)_{ik}) \quad (i \in N_{i*}(\delta), k = 1, \cdots, m) \tag{4}$$

여기에서 $W(old)_{ik}$는 조절되기 전의 연결강도이고, $W(new)_{ik}$는 조절된 후의 연결강도이다.

Kohonen의 SOFMs 학습 알고리즘은 다음과 같다.

절차 1 : 연결강도를 초기화한다. 학습률과 이웃의 범위를 정한다.
절차 2 : 입력층에 하나의 입력데이터를 입력한다.
절차 3 : 입력데이터와 연결강도 사이의 거리를 식 (1)과 같이 계산한다.
절차 4 : 하나의 승자노드를 결정한다.
절차 5 : 학습규칙에 따라 연결강도를 식 (4)와 같이 조절한다.
절차 6 : 이웃범위와 학습률을 감소시키면서 이웃범위가 승자노드 자신이 될 때까지 절차 2에서 절차 5까지 반복한다.

3. 제안하는 알고리즘

Kohonen의 SOFMs은 초기에 임의로 주어진 연결강도 및 학습률로 인하여 수행시마다 조금씩이지만 해의 변동이나 학습시간이 오래 걸리는 몇 가지 문제를 가지고 있다. 이와 더불어 초기에 임으로 결정된 출력노드의 연결강도는 해의 질에 많은 영향을 미치고 있다. 제안하는 SOFMs은 1차원 선형(linear) 출력층으로 이루어져 있다. 1차원 출력노드의 구조는 그룹을 형성하는 절차는 단순하나 초기에 임의로 선정된 연결강도에 따라 해의 질에 많은 영향을 받는다. 이와 같은 문제를 극복하기 위해 초기에 임의로 선정된 출력노드의 연결강도에 대한 유클리디언 거리를 계산하고, 이것을 바탕으로 출력노드를 오름차순으로 정렬한다. 이렇게 한 후 입력데이터에 대하여 각각의 승자노드가 결정된 후 직전의 노드와 직후의 노드가 이웃노드로서 승자노드와 함께 연결강도를 갱신하게 되면 초기에 임의로 주어진 출력노드별 연결강도를 바탕으로 승자노드를 결정하고, 이 노드의 직전노드와 직후노드를 이웃노드로 결정하는 학습방식과 다르다. 전통적인 방식은 이웃의 범위를 줄여가면서 입력데이터에 대해 학습하는 동안에 서로 다른 승자노드를 선정하는 결과를 가져와서 오히려 학습시간을 증가시킨다. 반면, 식 (5)와 같이 각 출력노드의 연결강도 합인 유클리디언 거리(Euclidean distance)를 구하고 이를 바탕으로 각 출력노드의 유클리디언 거리를 오름차순으로 정렬한다. 이렇게 구성된 출력노드에 입력데이터를 대응하고, 학습을 진행함으로써 일정한 간격으로 특정한 출력노드를 승자노드로 선정하여 입력데이터의 분포와 유사한 형태로 그룹을 형성한다.

$$W_i = |w_{i1}| + |w_{i2}| + \ldots + |w_{im}| \quad i = 1, 2, \cdots, n \qquad (5)$$

제안하는 SOFMs에서 출력노드의 개수는 입력데이터의 개수보다 크게 설정함으로써 출력노드의 분포를 입력데이터의 분포와 유사한 형태로 펼쳐 놓을 수 있도록 하였다. 학습과정에서 초기 이웃범위의 절반이 되는 시점에서 그때까지 승자노드가 되지 못한 출력노드의 연결강도는 식 (4)와 같이 조절하지 않는다. 이와 같이 학습이 진행된 입력데이터가 출력노드에 유사한 순서대로 정렬되면 출력노드간의 연결강도가 가장 큰 지점을 선형적으로 분리하여 원하는 그룹을 형성할 수 있다. 이것은 학습과정에서 발생할 수 있는 문제점뿐만 아니라 학습이후에 그룹을 형성하는 과정을 단순하게 처리할 수 있다.

제안하는 클러스터링 알고리즘은 출력노드의 개수, 학습률, 그리고 이웃범위와 같은 매개변수를 결정한다. 출력노드의 개수는 적을수록 학습시간이 짧아지므로 입력데이터가 출력노드에 충분히 펼쳐질 수 있는 최소의 출력노드 개수를 설정한다. 경험적 방법에 의하면 출력노드 개수는 입력데이터 개수의 두 배 이상으로 정하면 학습과정에서 발생할 수 있는 결함을 안전하게 극복할 수 있는 가장 좋은 해를 구할 수 있음이 나타났다. 초기에는 모든 출력노드를 이웃으로 설정하고, 시간이 지남에 따라 그 범위를 줄여나가고 이웃범위가 자기 자신일 때 알고리즘을 중지한다. 초기 학습률 즉 $\alpha(0)$는 경험적으로 좋은 결과를 제공하는 0.4로, 초기 이웃의 범위는 출력노드의 개수와 같은 수로 정한다. 각 입력데이터가 사상된 출력노드 i와 $i+1$연결강도간의 거리를 유클리드 거리를 이용하여 선형적으로 분리한다. 여기에서 i와 $i+1$은 이웃한 출력노드의 번호이다. 사상된 이웃한 출력노드의 연결강도 거리를 유클리드 거리로 사용하는 이유는 연결강도가 입력데이터와 유사한 확률분포함수 형태로 구성되기 때문에 단지 이웃한 출력노드의 연결강도 거리만을 고려하면 된다.

제안하는 클러스터링 알고리즘의 절차는 다음과 같다.

절차 1 : SOFMs의 구조(출력노드의 형태, 입·출력노드의 개수)를 초기화한다.

절차 2 : 각 연결선에 강도를 초기화 하고, 초기 학습률 $\alpha(0)$ 및 학습함수 $\alpha(t)$ 그리고 초기

이웃범위를 정한다.

절차 3 : 출력노드별 연결강도에 대한 유클리디언 거리를 구한다. 유클리디언 거리의 크기에 따라 오름차순으로 출력노드 $O(k)$를 정렬한다.

절차 4 : 각 입력데이터에 대하여 출력노드의 연결강도와 입력데이터 사이의 거리를 식 (1)과 같이 계산한다. 그 중에서 가장 짧은 노드를 승자노드로 결정한다.

절차 5 : 승자노드로부터 일정한 범위 내 위치한 이웃노드의 연결강도를 식 (4)와 같이 조절한다.

절차 6 : 이웃의 범위는 1 만큼, 학습률은 $\alpha(t)=(1-t/4950)\times\alpha(t-1)$만큼 감소시키면서 최종적으로 이웃범위가 승자노드 자신이 되거나 또는 학습률이 0가 될 때까지 절차 4에서 절차 5까지 반복한다. 단, 이웃범위가 초기 이웃의 범위에 절반이 되는 시점이후에는 그 동안 한번도 승자노드가 되지 못한 출력노드는 학습을 시키지 않는다.

절차 7 : 각 입력데이터를 가장 가까운 출력노드에 사상시킨다.

절차 8 : 입력데이터가 대응된 이웃한 출력노드 u와 v 간의 연결강도 거리 $WD(u,v)$를 식 (6)과 같이 계산한다.

$$WD(u,v)=|w_{u1}-w_{v1}|+|w_{u2}-w_{v2}|+\cdots |w_{um}-w_{vm}| \quad u,v=1,2,\cdots,n \tag{6}$$

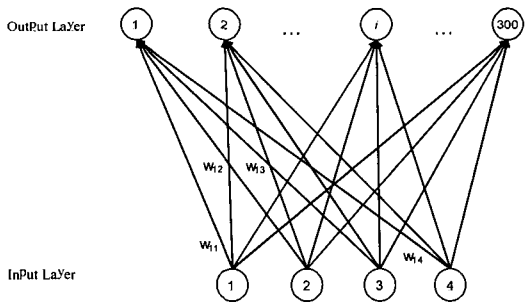
절차 9 : 절차 8에서 계산된 $WD(u,v)$를 내림차순으로 정렬하고, 이중에서 상위 $c-1$개 선택하여 c개의 그룹을 형성한다(단, 동률이 발생하는 경우 임의로 $c-1$개를 선정한다.).

4. 수치예제

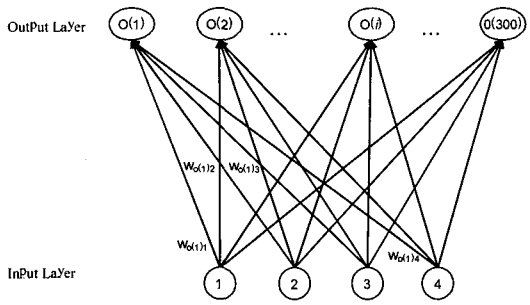
본 연구에서 사용한 데이터는 Anderson의 IRIS 데이터와 TSPLIB 데이터를 사용하였다. 특히, Anderson의 IRIS 데이터는 4차원(Petal Width, Petal Length, Sepal Width, Sepal Length) 속성을 가지는 150개 샘플 데이터로 구성된다. 세 개의 클러스터(Iris Setosa, Iris Verisiclolr, Iris Verginica)는 각각 50개의 데이터로 구성되어 있다. IRIS 데이터는 무감독학습을 이용한 클러스터링 알고리즘으로는 11~17개의 오 분류가 나타난다고 알려져 있다[14].

본 연구에서 사용하는 IRIS 데이터에 대하여 제안하는 클러스터링 알고리즘을 적용하면 다음과 같다.

절차 1 : 본 예제에서 사용하는 SOFMs의 구조는 [그림 3]과 같이 출력노드의 구조는 선형이고, 입력노드와 출력노드의 개수가 각각 4,300이다. 여기에서 출력노드의 구조는 선형이고, 출력노드의 개수는 입력데이터의 2배로 한다. 입력노드의 개수는 4개이다.



[그림 3] 제안하는 SOFMs의 구조



[그림 4] 정렬된 연결강도를 가지는 SOFMs의 구조

절차 2 : 첫 번째 출력노드의 연결강도는 W_1={0.5882, 0.2500, 0.2200, 0.1872}이고, 두 번째 출력노드의 연결강도는 W_2={0.9232, 0.4950, 0.8613, 0.7534}이고, 19번째 출력노드의 연결강도 W_{19}={0.0145, 0.2887, 0.0584, 0.0835}이고, 300번째 출력노드의 연결강도는 W_{300}={0.7083, 0.8935, 0.8302, 0.0759}이다. 초기 학습률은 0.4로 초기화하고, 학습함수는 $\alpha(t)=(1-t/4950)\times\alpha(t-1)$이다.

절차 3 : 첫 번째 출력노드 연결강도 합은 1.245이고, 두 번째 출력노드 연결강도 합은 2.5468이고, 19째 출력노드 연결강도 합은 0.4453이고, 300번째 출력노드 연결강도 합은 2.5082이다. [그림 4]와 같이 연결강도의 합이 오름차순으로 정렬된 출력노드의 정렬순서 O(k)는 19, 48, …, 44, 89이다.

절차 4 : 첫 번째 입력데이터(0.02, 0.14, 0.33, 0.5)에 대하여 첫 번째 출력노드의 연결강도 $W_{o(1)}$과의 거리를 식 (1)과 같이 계산하면 0.2748이다. 출력노드의 연결강도 $W_{o(1)}$, $W_{o(2)}$, …, $W_{o(300)}$에 대하여 계산하면 0.6902, 1.3757, …, 1.6861이고, 가장 짧은 출력노드 $W_{o(1)}$를 승자노드라고 한다.

절차 5 : 반경 300에 있는 모든 출력노드의 연결강도를 식 (4)와 같이 조절한다.

절차 6 : 이웃의 범위를 1만큼 줄이고 $t=0$인 경우 즉 초기 학습률 $\alpha(0)$을 0.4로 $t=1$인 경우 $0.4\times(1-(1/4950))$으로 감소시키면서 이웃 범위가 승자노드 자신이 될 때까지(반경이 0이 될 때 까지) 절차 4에서 절차 5까지 반복한다.

절차 7 : 각 입력데이터를 가장 가까운 출력노드에 사상시키면 <표 1>과 같다.

절차 8 : <표 2>에서 각각의 출력노드와 출력노드 간의 연결강도 거리를 식 (6)과 같이 계산하면 WD(1, 5)=0.0993이고, …, WD(56, 75)=2.8132, …, WD(182, 191)=0.4176, …, WD(299, 300)=0.0207이다.

절차 9 : 연결강도사이의 거리가 가장 큰 출력노드와 출력노드 구간은 56과 75이고, 그 다음으로는 182와 191이다. 따라서 이 출력노드사이에서 입력 데이터들 간의 상이성이 존재한다. 각각의 값은 WD(56, 75)=2.8132, WD(182, 191)=0.4176이다. 일직선으로 펼쳐진 출력노드에서 출력노드 56번과 75번 사이를 분리하고, 또한 출력노드 182번과 191번 사이를 분리함으로써 출력노드 1번부터 출력노드 56번까지는 첫 번째 그룹, 75번부터 182번까지는 두 번째 그룹, 그리고 191번부터 300번까지가 세 번째 그룹이다. 정리하면 3개의 그룹은 <표 3>과 같다.

5. 실험결과 및 분석

본 연구에서 제안하는 알고리즘의 성능을 평가하기 위해 기존 연구에서 인용되는 IRIS 데이터와 TSPLIB 데이터를 대상으로 실험하였다. <표 3>과 같이 오 분류된 데이터는 그룹 2에서 1개, 그룹 3에서 3개 총 4개의 오 분류가 나타남을 알 수 있다. 이 결과는 <표 5>에 나타난 기존의 클러스터링 알고리즘보다 좋은 결과를 제시하였다.

본 연구에서 제안하는 SOFMs의 출력노드 구조는 1차원 선형구조를 하였으나 초기에 임의로 설정되는 연결강도로 인하여 좌측에서 우측으로 출력노드의 연결강도 합이 크기 순으로 정렬되지 못하였다. 이것을 수정하기 위하여 출력노드의 연결강도 합을 기준으로 좌측에서 우측으로 오름차순으로 정렬하였다. 또한, IRIS 데이터와 TSPLIB 데이터의 근사 최적 해를 구할 수 있는 출력노드의 개수를 결정하기 위해 출력노드 개수는 3개에서 시작하여 입력데이터의 4배까지 증가시키면서 실험한 결과 출력노드의 개수가 3개에서 입력데이터의 2배 미만에서는 수행시마다 다소 다른 결과를 보였으나 입력데이터의 2배 이상일 경우는 동일한 결과를 나타냄을 확인하였다.

<표 4>와 같은 실험에 의하면 오분류의 개수는

초기 학습률 및 학습함수, 이웃범위 등에 민감한 반응을 보였다. 여기에서 학습함수를 $1-t/5000$로 한 경우에는 기존 오분류와 비교하여 최저인 4개의 오분류를 발견하였으나 시행 시마다 약간 다른 결과를 보여주었다. 그러나 학습함수를 $1-t/4950$로 한 경우에는 안정적으로 최소의 오분류를 보장하였다. 이와 더불어 출력노드의 개수가 입력데이터의 2배 미만인 경우 학습과정에서 발생할 수 있는 여러 가지 결함이 발생할 가능성이 큰 것으로 추정된다. 제안하는 알고리즘은 초기 학습률은 0.4로, 학습함수는 $(1-t\times(1/4950))$로 하였다. 초기 이웃의 범위는 모든 출력노드의 연결강도를 변경할 수 있도록 반경을 300으로 하였다. 학습이 진행되어 초기 이웃의 범위 절반인 150 이후에는 그때까지 승자노드가 되지 못한 출력노드는 연결강도를 조절하지 않는다.

〈표 1〉 정렬된 출력노드에 대응된 입력데이터의 번호

1)	2)	1)	2)	1)	2)	1)	2)
1	6, 11, 15, 16 17, 19, 33, 34	54	26	144	79, 86	236	112
		56	25	147	98	237	117
5	37, 49	75	99	150	69, 88	238	111
8	20, 45, 47	77	58	151	74	240	129
10	21	78	94	153	92	241	133
12	22, 32	80	61	154	64	245	116, 149
15	28	83	80	157	75	250	142, 146
18	18	85	82	166	52, 76	253	137
19	1,5, 29, 44	89	81	167	66	260	105
21	24, 40	93	65	169	59	261	101
22	27, 41	96	70	170	55	262	141, 145
23	8, 38	97	54	175	51, 77, 87	263	113
27	50	98	60	176	53, 57	264	140
29	12	100	90	182	78	266	125
31	36	102	63	191	73, 134	267	109
32	23	105	83, 93	196	124, 127	271	121
34	7	111	68	202	128, 139	272	144
39	3	114	107	204	71	280	103
40	48	115	91	205	120	283	130
41	14, 43	116	95	206	84	284	126
42	39	119	100	211	150	286	110
44	4, 9, 13	121	89	213	122	293	131
47	2, 10, 46	125	97	214	102, 114, 143, 147	294	108
48	42	126	85, 96	218	115	299	136
49	35	128	56, 67	219	135	300	106, 118, 119, 123, 132
50	30	134	62	227	104		
52	31	136	72	232	138, 148		

주) 1) 출력노드의 번호
2) 입력데이터의 번호

〈표 2〉 최종적으로 정렬된 출력노드의 연결강도

출력노드의 번호	연결강도	입력데이터의 번호	$WD(u,v)$
1 ($u=1$, $v=5$)	5.32215, 3.76604, 1.49669, 0.273204	6, 11, 15, 16, 17, 19, 33, 34	0.0993
5	5.2799, 3.71155, 1.49921, 0.273071	37, 49	
...			
56 ($u=56$, $v=75$)	4.79075, 3.00316, 1.72428, 0.322601	25	2.8132
75	5.18422, 2.58289, 3.12393, 0.922425	99	
...			
182($u=182$, $v=191$)	6.4114, 2.91433, 4.78981, 1.56679	78	0.4176
191	6.2709, 2.84233, 4.90216, 1.65957	73, 134	
...			
299($u=299$, $v=300$)	7.45747, 3.13188, 6.30752, 2.08663	136	0.0207
300	7.46556, 3.13149, 6.31746, 2.08433	106, 118, 119, 123, 132	

〈표 3〉 3개의 그룹에 할당된 입력데이터

그룹번호	입력데이터의 번호
1	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50
2	51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 72, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 107
3	71, 73, 84, 101, 102, 103, 104, 105, 106, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150

〈표 4〉 학습함수(초기 학습률 포함)의 변화에 따른 오 분류의 개수 변화(반복횟수 : 30)

학습함수 \ 초기 학습률	0.1	0.2	0.3	0.4
1-t/3000	17	17	17	17
1-t/4000	17	17	15	15
1-t/5000	15	15	9	4
1-t/6000	17	17	15	12

<표 5>는 Anderson의 IRIS 데이터에 대하여 제안하는 클러스터링 알고리즘과 기존 클러스터링 알고리즘의 해를 비교한 결과이다. 제안하는 클러스터링 알고리즘은 Pal et al.[14]이 구한 오 분류 17개와 Karayiannis[9]이 구한 오 분류 15개 보다 더 좋은 오 분류 4개를 갖는 해를 구하였다. Pal et al.[14]에 의하면 기존의 무감독학습을 이용한 클러스터링 알고리즘에서는 적어도 11~17개의 오 분류를 생성한다고 하였다. <표 5>에서 제안하는 클러스터링 알고리즘의 오 분류 개수는 4개로 기존의 알고리즘에서 제시하는 11~17보다 더 적은 오 분류를 나타낸다.

〈표 5〉 오 분류의 개수 비교

문제의 출처	알고리즘의 출처	오 분류된 입력데이터의 개수
Anderson's IRIS Data Set[2]	Proposed algorithm	4
	Tasao et al.[16]	11
	Karayiannis[9]	15
	Pal et al.[14]	17

<표 6>은 TSPLIB 데이터에 대하여 제안하는 클러스터링 알고리즘의 해와 최적 해를 비교한 결과이다. 제안하는 클러스터링 알고리즘은 최적 해로부터 6.5% 범위 내에 있는 근사 최적 해를 발견하였다. 또한 이 값은 전역 탐사 알고리즘의 초기해로

사용하여 빠른 시간 내 효율적으로 최적 해를 구할 수 있다.

〈표 6〉 TSPLIB문제에 대한 실험결과

문제	도시의 개수	최적 이동 거리	제안하는 알고리즘	
			이동 거리	최적해로 부터 %내
eil51	51	426	435.51	2.2
st70	70	538	561.96	4.5
eil76	76	629	670.03	6.5
eil101	101	675	685.75	1.6

6. 결 론

본 연구에서는 IRIS 데이터의 분류 및 TSPLIB의 최단 이동경로를 발견하는 효율적인 클러스터링 알고리즘을 제시한다. 본 연구의 특징은 SOFMs의 구조와 파라미터 연구에 있다. 제안하는 SOFMs의 구조는 1차원 선형구조이다. 출력노드의 개수는 입력노드의 2배로 설정하였다. 이와 같은 출력노드에 연결강도를 임의로 부여하고, 출력노드별 연결강도 합의 크기에 따라 오름차순으로 정렬하였다. 학습이 진행되고 이웃노드의 범위가 초기 이웃노드 범위의 절반이 되는 지점에서 승자노드가 되지 못한 출력노드의 연결강도는 조절하지 않는다. 학습이 완료된 입력데이터가 출력노드에 유사한 순서대로 정렬되면 출력노드간의 연결강도 차이가 가장 큰 지점을 선형적으로 분리하여 원하는 그룹을 형성할 수 있다.

클러스터링 알고리즘은 초기 학습률 및 학습함수 그리고 이웃범위와 같은 매개변수를 어떻게 결정하느냐에 따라 성능이 좌우된다. 초기 학습률은 클수록 학습시간이 줄어든다. 본 연구에서는 초기 학습률을 0.4, 학습함수를 $\alpha(t)=(1-t/4950)\times\alpha(t-1)$로 하였다. 반면 출력노드의 개수와 이웃범위는 적을수록 학습시간이 짧아지나 입력데이터가 출력노드에 충분히 펼쳐질 수 있는 최소의 출력노드 개수 및 이웃범위를 설정하는 것이 매우 중요하다. 경험적 방법에 의하면 출력노드 개수는 입력데이터 개수의 두 배 이상으로 정하면 학습과정에서 발생할 수 있는 결함을 안전하게 극복할 수 있는 가장 좋은 해를 구할 수 있음이 나타났다. 이웃범위는 초기에는 모든 출력노드를 이웃으로 설정하고, 시간이 지남에 따라 그 범위를 줄여나가고 이웃범위가 자기 자신일 때 알고리즘을 중지한다.

본 연구에서는 잘 알려진 IRIS 데이터와 TSPLIB 데이터를 가지고 실험하였다. 제안하는 클러스터링 알고리즘은IRIS 데이터를 가지고 실험한 결과 이제까지 알려진 기존의 무감독학습을 이용한 클러스터링 알고리즘이 11~17개의 오 분류를 생성하는 데 비해 4개의 오 분류를 생성함으로써 더 좋은 해를 구하였다. 또한 TSPLIB 데이터의 이동경로를 발견하였는데 모두 근사 최적 해를 구하였다. 제안하는 알고리즘은 복잡한 연산을 사용하지 않기 때문에 실시간으로 사용이 가능하며 변화하는 상황에 유연하게 적용할 수 있는 장점도 가지고 있다.

참 고 문 헌

[1] Aldenderfer, M.S. and R.K. Blashfield, *Cluster Analysis*, Saga Publications, London, 1984.

[2] Anderson, E., "The IRIS's of the Gaspe Peninsula," *Bull. Amer. IRIS Soc.*, Vol.59 (1939), pp.2-5.

[3] Berry, M.J.A. and G. Linoff, *Data Mining Techniques for Marketing, Sales, and Customer Support*, John Wiley & Sons, New York, 1997.

[4] Berry, M.J.A. and G.S. Linoff, *Data Mining Techniques for Marketing, Sales, and Customer Relationship Management*, John Wiley & Sons, New York, 2004.

[5] Bezdek, J.C., *Pattern Recognition with Fuzzy Objective Function Algorithms*, Plenum, New York, 1981.

[6] Everitt, B.S., *Cluster Analysis*, Edward Arnold, London, 1993.

[7] Everitt, B.S., S. Laudau, and M. Leese, *Cluster Analysis*, Edward Arnold, London, 2001.

[8] Huntsberger, T.L. and P. Ajjimarangsee, "Parallel Self-Organizing Feature Maps for Unsupervised Pattern Recognition," *International Journal of General Systems*, Vol.16, No.4(1990), pp.357-372.

[9] Karayiannis, N.B., "A Methodology for Constructing Fuzzy Algorithms for Learning Vector Quantization," *IEEE Trans. Neural Networks*, Vol.8, No.3(1997), pp. 505-518.

[10] Kohonen, T., *Self-Organization and Associative Memory,* Springer, Berlin, 1984.

[11] Kohonen, T., *Self-Organization and Associative Memory*, Springer, Berlin, 1988.

[12] Kohonen, T., *Self-Organizing Maps,* Springer, Berlin, 1997.

[13] Mangiameli, P., S.K. Chen, and D. West, "A Comparison of SOM Neural Network and Hierarchical Clustering Methods," *European Journal of Operational Research*, Vol. 93, No.2(1996), pp.402-407.

[14] Pal, N.N., J.C. Bezdek, and E.C.K. Tasao, "Generalized Clustering Networks and Kohonen's Self-Organizing Scheme," *IEEE Trans. Neural Networks*, Vol.4, No.4(1993), pp.549-551.

[15] Reinelt, G., "TSPLIB : A Traveling Salesman Problem Library," *ORSA Journal on Computing*, Vol.3(1991), pp.376-384.

[16] Tasao, E.C.K., J.C. Bezdek, and N. N. Pal, "Fuzzy Kohonen Clustering Networks," *Pattern Recognition*, Vol.27, No.5(1994), pp.754-757.