2005 한국경영과학회/대한산업공학회 춘계공동학술대회
2005년 5월13일~14일, 충북대학교

인공데이터첨가를 통한 SOM의 quantization error 감소
Error reduction by adding artificial data in SOM

김승택 조성준
서울대학교 산업공학과

Abstract

자기조직화지도(Self Organizing Map, SOM)는 비지도 신경망으로서 고차원의 입력공간을 위상적관계를 유지시키면서 저차원으로 사영 시킬 수 있는 특징을 갖고 있다. SOM은 패턴인식과 자료압축/재생 등 여러 분야에서 유용하게 활용될 수 있으며 특히 고차원 자료의 시각화 방법으로 많은 관심을 받고 있다. 본 연구에서는 SOM의 quantization error를 줄이기 위한 목적으로 인공데이터를 생성시켜 학습에 이용하는 방법을 제시한다. 이는 특히 데이터가 부족한 상황에서 SOM을 학습시켜야 할 때 유용하게 적용될 수 있을 것으로 기대된다.

1. 서론

SOM(Self Organizing Map)은 신경망의 일종으로 1970년대 후반에서 1980초에 걸쳐 핀란드의 Kohonen에 의해 고안되었다. 다른 신경망들과는 달리 SOM은 비지도 학습을 수행하는 모형이다.

SOM은 두 가지 중요한 특징을 가지고 있는데 하나는 위상적 관계를 유지하면서 고차원 입력공간을 1차원 또는 2차원과 같은 저차원 공간으로 맵핑할 수 있다는 것이며 다른 하나는 한정된 수의 유닛으로 입력공간의 분포를 근사 또는 축약시킬 수 있다는 점이다. 이러한 시각화, 축약의 특성으로 인해 SOM은 자료 분석, 패턴 인식, 통신 등 다양한 분야에서 활용될 수 있다[1].

SOM 알고리즘은 output map상의 노드들이 입력공간의 분포를 잘 반영할 수 있게 이들의 weight를 적절히 위치시키는 것을 목적으로 한다. 이 때 가중치들의 최적 위치를 결정하는 closed form solution이 존재 하지 않기 때문에 알고리즘은 iterative한 과정으로 진행이 되며 결과적으로 weight들이 주어진 학습데이터에 밀착하는 형태로 학습이 진행된다[2]. 따라서 일반적으로 SOM을 학습시킬 때는 모집단을 충분히 반영할 수 있는 많은 양의 데이터가 필요하다.

하지만 실제 상황에서는 그 정도로 충분한 양의 데이터를 갖추고 있기가 어렵기 때문에 한정된 학습데이터만을 가지고 학습을 시켜야 하며 따라서 데이터모집단의 분포를 정확하게 반영하도록 SOM을 학습시키는 것은 사실상 어려운 경우가 많다.

본 연구에서는 SOM을 학습할 때 인공의 데이터를 첨가하는 방법으로 부족한 데이터를 어느 정도 보완할 수 있는 방법을 제안한다. 첨가된 인공데이터는 데이터셋의 밀도를 높임과 동시에 원데이터에 대한 학습과정에서 SOM이 과도하게 원데이터에 적합 되는 것을 막음으로써 좀더 향상된 결과를 낼 수 있도록 해준다.

2. 관련연구

인공의 데이터를 활용하여 모형의 성능을 높이려는 시도는 새로운 것은 아니다. Jang & Cho(2000)은 가상의 데이터를 이용한 앙상블 학습알고리즘을 제안한 바 있고 G.An.(1996)은 신경망의 학습에 인공데이터를 첨가해 예측 성능을 향상시키는 방법을 연구하기도 하였다. 하지만 이러한 연구들은 지도학습에 있어서 예측정확도를 높이기 위해 인공데이터를 사용했던 것으로 본 연구의 방향과는 거리가 있다.

SOM의 성능을 향상시키기 위한 시도로는 앙상블 SOM이 대표적이다. Yuan & Zhou(2004)가 그들의 논문에서 SOM을 앙상블하여 이미지를 효과적으로 구분하는 방법을 제안한 바 있으며 Cho(2000)도 SOM을 앙상블하여 성능향상을 시도하였다. 하지만 인공의 데이터를 이용한 SOM의 성능향상 시도는 아직까지 소개된 바가 없다.

2. SOM 알고리즘

SOM 네트워크는 두 개의 층으로 구성되어 있다. 입력층(input layer)과 코호넨층(Kohonen layer)이 바로 그것이다. 입력층과 코호넨층은 서로 완전 연결(fully connected)되어 있다. 입력층은 주어진 변수들과 같은 차원으로서 학습데이터를 받아들이는 역할을 하며 코호넨층은 SOM의 핵심적인 부분으로, 각각의 입력패턴을 양자화(quantization)시켜서 1차원 또는 2차원과 같은 저차원의 격자(output grid)로 내보내는 기능을 담당한다. 여기서 양자화란 각각의 입력 패턴들을 output grid의 노드들에 매칭시키는 것을 의미한다. 매칭 과정은 입력패턴과 각 노드 가중치 사이의 거리를 계산하여 가장 가까운 노드와 입력패턴을 대응시키는 식으로 이루어진다. 이 과정에서 입력공간 상의 위상적 관계(topological relation)는 output grid상에 그대로 유지되어 나타나게 됨으로써 고차원자료의 시각화와 같은 SOM특유의 기능을 수행할 수 있게 된다.

SOM의 학습 알고리즘은 다음과 같다.

1) 노드의 가중치를 임의의 값으로 초기화.
2) t번째 입력패턴 $x(t)$.
3) 입력패턴과 노드 가중치 사이의 거리를 계산하고 승자 노드(winner node)를 결정 즉,
$\forall i, \quad \|\boldsymbol{x}(t) - \boldsymbol{m}_c(t)\| \le \|\boldsymbol{x}(t) - \boldsymbol{m}_i(t)\|$
$m_c(t)$: winner node
4) 코호넨 학습법칙에 의해 노드들의 가중치를 갱신시킴.

$$\boldsymbol{m}_i(t+1) = \boldsymbol{m}_i(t) + h_{c(x),i}(\boldsymbol{x}(t) - \boldsymbol{m}_i(t))$$

$$h_{c(x),i} = \alpha(t)\exp\left(-\frac{\|\boldsymbol{r}_i - \boldsymbol{r}_c\|^2}{2\sigma^2(t)}\right)$$

$h_{c(x),i}$ 는 이웃함수(neighborhood function)라고 불린다. 위에 보이는 바와 같이 이웃함수로는 흔히 가우시안커널이 사용된다. 이 함수는 소위 말하는 종모양(bell-curve)함수로서 승자

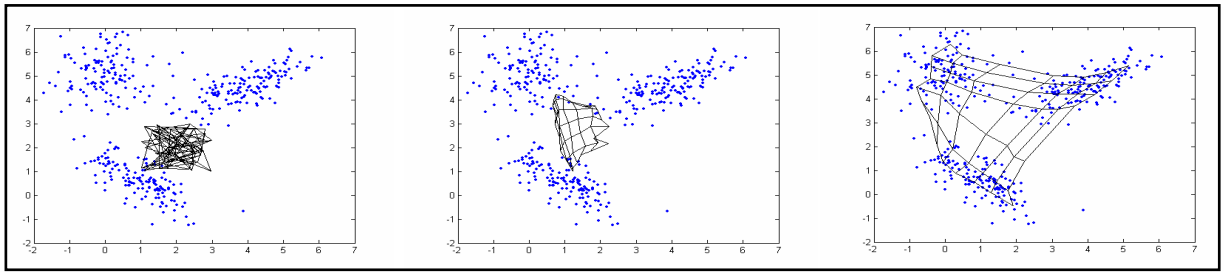

그림 1. SOM의 학습과정

노드와 가까이 있는 노드일수록 가중치 갱신이 큰 폭으로 일어나게끔 하는 역할을 수행한다. 여기서 $\|r_i - r_c\|^2$ 는 output grid 상에서 승자노드 m_c 와 비승자노드 m_i 사이의 거리를 의미한다. $\alpha(t)$ 는 0에서 1사이의 학습률을 나타내며 $\sigma(t)$ 는 이웃함수의 이웃반경으로써 가중치 갱신 시에 승자노드와 함께 갱신이 될 이웃노드의 범위를 조절한다. 이 두 값은 모두 t에 대해 단조감소 하는 값이다. 즉 학습이 진행될수록 학습률과 이웃반경을 감소시켜 수렴을 용이하게 한다. 이웃반경과 학습률은 t에 따라 감소하는 함수면 어떤 것이든 무방하나 Ritter et al.은 그들의 연구에서 다음과 같은 함수를 제안한 바 있다[3].

$$\sigma(t) = \sigma_i(\sigma_f / \sigma_i)^{t/t_{\max}},$$
$$\alpha(t) = \alpha_i(\alpha_f / \alpha_i)^{t/t_{\max}}$$

여기서 $t_{\max}$ 는 SOM의 학습과정 동안 적용될 데이터의 총 수를 나타낸다. σ_i 와 σ_f 는 파라메터로 각각 $\sigma(t)$ 의 최대값과 감소속도를 제어하는 역할을 한다. Ritter et al.이 제안한 함수를 사용하면 이러한 파라메터들을 잘 조합하여 사용자의 요구에 부합하는 이웃반경을 정교하게 설계할 수 있다는 장점이 있다. 이는 학습률 설정에 있어서도 마찬가지다. 본 연구에서도 Ritter et al.의 함수를 사용하기로 한다.

그림1 은 SOM의 학습과정을 보여주고 있다. 점들은 학습데이터이며 가운데 그물처럼 보이는 것은 노드들의 가중치벡터들이다. 가중치벡터간 연결선은 각 노드 간의 이웃관계를 나타낸다. 처음에 마구 엉켜있던 가중치들이 학습이 진행됨에 따라 이웃관계를 유지 하면서 자리를 잡아나가고 있다.

3. 제안하는 알고리즘

본 연구에서 제안하는 알고리즘은 기본적으로는 인공의 데이터를 생성해 내는 알고리즘이다. SOM의 본체를 수정하지는 않으며 생성된 인공데이터를 SOM의 학습과정에 포함시켜서 에러를 줄이려는 것이 목적이다. 간단히 설명하자면, 주어진 학습데이터에 SOM을 학습시킨 후 SOM의 노드의 가중치벡터 주위에 인공의 데이터를 생성시켜 데이터의 밀도를 높인 다음 이를 가지고 SOM을 재학습 시킴으로써 smoothing 효과에 의한 에러감소를 꾀하고자 하는 것이다. 구체적인 알고리즘은 다음과 같다.

1) N←0, D_N←초기에 주어진 학습데이터, D_N으로 SOM을 학습한다.
   이 때 epoch을 충분히 주어 SOM이 데이터셋에 완전히 적합 되도록 한다.
2) 학습된 SOM에 다시 한 번 D_N을 적용시켜 SOM의 각 노드에 매칭되는 데이터와 노드별 승리 횟수를 기록한다, N←N+1
3) 각 노드의 승리 횟수에 비례시켜 노드의 가중치벡터 주위에 인공데이터를 생성시킨다→A_N. 이 때 인공데이터들은 가중치벡터를 중심으로 하는 정규분포로부터 생성한다. 이 때 필요한 공분산 행렬은 노드 별 receptive region내 데이터들을 근거로 산출한다. 단, region 내 데이터가 3개 미만일 경우엔 기존 데이터를 복사하여 추가한다.
4) D_N←$D_{N-1} \cup A_N$
5) D_N으로 SOM을 학습
6) If N < predetermined number, goto 2
   Else end

충분한 epoch으로 SOM을 주어진 학습데이터셋에 밀착시키는 것은 학습된 SOM을 근거로 새로이 생성될 인공데이터가 학습데이터의 분

포에서 크게 벗어나지 않게 하기 위한 것이다. 노드의 가중치 주변에 인공데이터가 생성되므로 노드 가중치들이 최대한 학습데이터의 분포를 잘 반영할 수 있어야 하는 것이다. 본 연구에서는 100번의 epoch을 적용하였다.

초기 학습이 완료된 후에는 노드에 매칭되었던 데이터들에 대한 기록을 가지고 인공데이터를 생성시킨다. 노드에 매칭되었던 데이터들을 기록하여 공분산행렬을 산출한 다음 이를 이용하여 가중치벡터를 중심으로 하는 정규 분포로부터 인공데이터를 만들어내는 것이다(그림2). 단, 노드에 너무 적은 수의 데이터가 매칭되고 있을 경우에는 공분산행렬을 구하여 정규 분포를 추정해 내기가 어려우므로 매칭되었던 데이터를 그대로 복사하는 식으로 데이터를 생성시킨다. 데이터를 생성하는 과정은 모든 노드에 대해 개별적으로 이루어진다.

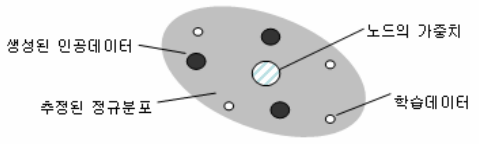

그림 2. 알고리즘에 의한 데이터 생성 과정

각 노드의 승리횟수에 비례하여 인공데이터를 생성시키는 것은 주어진 학습데이터의 밀도까지 반영하기 위함이다. 어떤 노드의 승리횟수가 다른 노드보다 많다는 것은 그 노드 주위에 그만큼 더 많은 학습데이터들이 분포하고 있었다는 것을 의미하므로 이러한 밀도 차이를 반영하여 데이터를 생성시키는 것이 합리적이다.

노드 승리횟수를 반영하기 위해 여기서는 노드 별로 (승리횟수/현재 루프 수)를 반올림 한 값을 새 데이터 생성 개수로 하였다. 승리횟수 값 자체를 생성시킬 데이터 수로 사용하는 방법도 생각할 수 있으나 그렇게 하면 루프가 반복될수록 데이터 셋의 크기가 지수적으로 증가하게 되므로 문제가 된다. 승리횟수를 루프 수로 나눈 값을 사용하면 매 루프에서 대략 초기 데이터갯수 만큼의 데이터만 생성된다.

SOM을 학습시키고 데이터를 생성시키는 과정은 미리 정해둔 횟수만큼 반복된다. 이 때 N번째 루프에서 생성된 인공데이터도 N+1번째 루프에서 데이터 생성 시에 활용된다. 반복횟수를 미리 정해두는 것은 실제문제 상황에서는 몇 번의 루프를 반복하는 것이 최적일지를 알 수 있는 방법이 없고 이는 또한 데이터 마다 다를 것이기 때문이다. 따라서 본 연구에서도 일반적인 반복횟수를 결정하지는 않는다. 단, 최소한의 일반 요건을 다음 절에서 실험과 함께 논의하기로 한다.

새롭게 생성된 데이터 셋으로 SOM을 재학습시킬 때에는 현재 학습된 상태의 SOM에 연속적으로 새로운 데이터를 학습시키는 경우와 SOM을 초기화 한 상태에서 처음부터 다시 학습을 시키는 경우의 두 가지 선택사항이 존재하는데 전자의 방법을 택하기로 한다. 몇 번의 시행착오 결과 연속적으로 학습시킬 경우가 더 안정적인 성능을 보임을 알 수 있었다. 연속적으로 학습 시킬 때에는 초기 학습된 SOM의 t를 유지 시키면서 t_max만을 다음과 같이 갱신해 주고 학습을 진행시키면 된다.

$$t_{\max} = t_{\max} + (size \times epoch2)$$

여기서 *size*는 새로운 데이터 셋의 크기이며 epoch2는 새 데이터셋에 적용할 epoch수 이다.

4. 실험

이번 절 부터는 위의 알고리즘을 실제로 데이터에 적용해 보면서 그 특성을 다각적으로 알아보고 다양한 데이터에 대해서도 적용해 보면서 일반성능을 측정한다. 먼저 세 가지 데이

터셋에 대해 몇 가지 실험을 행하여 속성을 파악한 후 그 결과를 근거로 필요한 파라메터를 고정시키고 더욱 많은 실제 데이터에 대해 실험해 봄으로써 이 알고리즘의 일반성능을 알아보도록 하겠다.

4.1 사용된 데이터

알고리즘의 특성을 파악하기 위해 사용한 실험용 데이터는 총 세 가지로 다음과 같다.

표 1. 사용된 데이터

	데이터수	차원	비고
3gauss	500	2	인공데이터
Abalone	500	8	UCI
Pima_indian	500	6	UCI

3gauss는 실험목적으로 인위적으로 만들어낸 2차원 데이터로 3가지 범주의 정규분포를 이루는 총 500개의 데이터이다. 각 범주 별로 100, 200, 200개씩으로 구성되어 있다. 각 범주의 평균, 공분산은 다음과 같다.

표 2. 3gaus 데이터 설명

Class	평균	공분산	개수
1	(0.1, 5)	$\begin{pmatrix} 0.5 & 0 \\ 0 & 0.6 \end{pmatrix}$	200
2	(1, 0.6)	$\begin{pmatrix} 0.5 & -0.3 \\ -0.5 & 0.7 \end{pmatrix}$	200
3	(4, 4.5)	$\begin{pmatrix} 0.4 & 0.2 \\ 0.5 & 0.6 \end{pmatrix}$	100

Abalone 데이터와 Pima_indian 데이터는 UCI machine learning repository[1]에 포함되어 있는 데이터이다.

4.2 파라메터 설정

map size는 실험에 쓰이는 데이터의 크기를 고려하여 임의로 정하였다. 주로 7×7 을 사용하였으며 50개 이하의 데이터로 실험 할 때엔 5×5 맵을 사용하였다.

이웃함수에 필요한 값들은 $\alpha_i = 0.6$, $\alpha_f = 0.005$, $\sigma_f = 0.5$ 로 각각 정하였으나 초기 이웃 반경을 결정하는 σ_i 는 map size에 따라 변화되어야 하므로 상황에 맞게 알맞은 값을 사용하여야 한다. 7×7 맵에서는 3으로 설정하였다.

4.3 평가 척도

SOM의 성능을 평가하는 척도로 여기서는 Quantization error를 이용한다. Quantization error를 나타내는 식은 다음과 같다.

$$E = \int \|x - m_c\|^r \, p(x)dx$$

X는 data sample 들을, m_c 는 x에 대응하는 노드의 가중치벡터이다. 결국, 학습이 잘 된 SOM은 가중치벡터들이 데이터들과의 거리가 최소가 되도록 자리 잡힌 SOM을 의미한다.

이 에러는 Full set data에 대해 산출한다. Full set data란 sampling된 학습데이터를 포함한 데이터 전체를 지칭한다. 문제 상황이 한정된 양의 주어진 데이터로 데이터 모집단을 잘 설명할 수 있는 SOM을 학습시키는 것이므로 데이터 모집단에 대해 error를 산출하는 것이 합리적이다.

4.4 결과의 평가

결과에 대한 평가는 주어진 데이터 만으로 학습했을 때의 에러(original error)와 알고리즘 적용 후 새 데이터 셋을 적용한 후의 에러를 비교하여 에러 감소 폭(%)을 측정하는 식으로 이루어진다. 주어진 데이터란 앞서 소개한 데이터셋에서 샘플링된 데이터가 된다. 이 때 original error는 충분한 학습을 통해 주어진 데이터로 얻어낼 수 있는 최소의 에러 이어야 한다. 100번의 epoch을 적용 한 후의 에러를

[1] http://www.ics.uci.edu/~mlearn/MLRepository.html

original error로 삼기로 한다.

5. 결과

5.1 루프 수에 따른 효과 측정

먼저, 루프 수에 따른 성과 정도를 알아본다. 세 가지 데이터에 대해 루프를 1부터 10까지 변화 시켜 가면서 향상 정도를 측정하였다. 각 루프 별로 50회의 반복 시행을 거쳤으며 각 시행마다 데이터는 독립적으로 random sampling 하여 사용하였다. 이 때 사용된 원데이터는 100개 씩이며 map size는 7×7 을 이용하였다. 그림3은 50회 반복 시행 중 에러가 감소한 경우를 성공이라고 했을 때 성공횟수를 표시한 그림이다. 그림4는 50회 반복 시행 동안 감소한 에러의 평균이며 그림5는 50회 시행 동안 최대, 최소 에러 감소율을 나타낸다. 에러 감소율에서 감소율이 음수인 것은 그만큼 에러가 증가한 것을 의미한다. 알고리즘 적용 후 에러가 오히려 증가하는 경우는 주로 원데이터에 노이즈가 많이 포함되어 있거나 전체 분포를 고르게 반영하지 못하는 데이터가 주어질 때이다. 이 알고리즘은 주어진 데이터를 기본 구조로 하여 그 밀도를 높이는 데에 목적이 있는 것으로 처음부터 데이터모집단과는 거리가 먼 데이터가 주어질 경우에는 새로 생성되는 데이터 역시 모집단에서 벗어나게 되므로 좋은 효과를 기대 할 수 없다.

그림을 보면 루프횟수 2회 일 때 세 데이터 모두 성공률이 안정적인 것으로 나타나고 있다. 루프 수가 증가하게 되면 성공률은 변화 없거나 감소하는 양상이 나타나며 최대 에러 감소율이 커짐과 동시에 최소 에러 감소율도 불안정하게 나타난다. 이는 루프가 진행될 수록 주어진 데이터 분포에서 벗어나는 데이터가 생성될 가능성이 증가하기 때문으로 판단된다. 이러한 데이터들이 양의 효과를 준다면 에러는 큰 폭으로 감소할 것이며 음의 효과를 준다면 큰 폭의 에러증가를 가져올 것이다. 결국 많은 수의 루프를 돌릴 때에는 큰 효과를 얻을 가능성이 높아지는 반면에 그만큼 위험이 존재함을 알 수 있다.

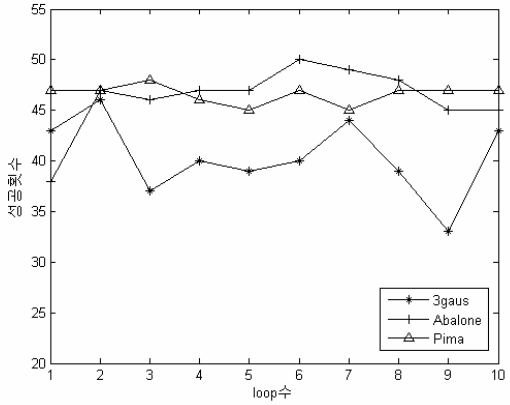

그림 3. 루프횟수에 따른 성공횟수 변화

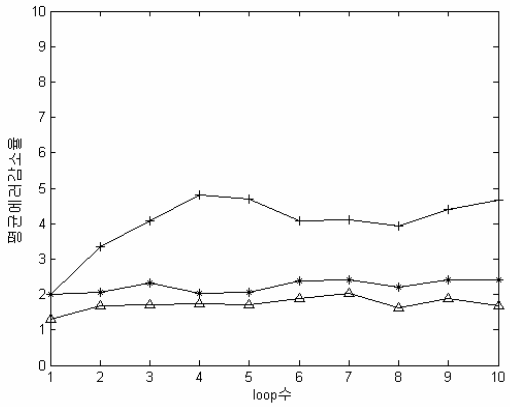

그림 4. 루프횟수에 따른 에러감소율의 평균

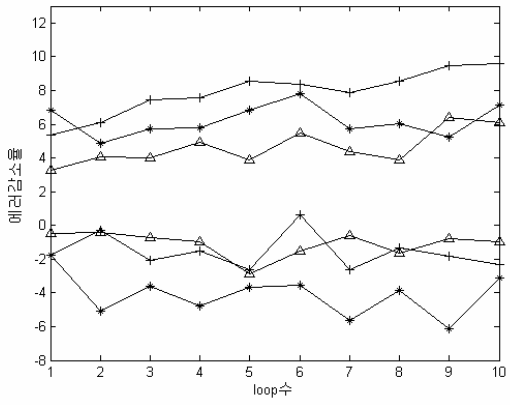

그림 5. 최대, 최소 에러감소율

결과를 보면 데이터마다 기대할 수 있는 성

과에도 차이가 있음을 알 수 있다. Abalone 데이터의 경우 알고리즘을 통해 현격한 효과를 볼 수 있음에 반해 3gaus데이터는 성공률이 불안정한 데다가 기대할 수 있는 에러 감소율도 2% 정도에 그치고 있다. Pima_indian 데이터의 경우에는 성공률은 매우 높으나 기대할 수 있는 효과는 낮다.

이러한 에러 감소율이 실질적으로 어느 정도의 효과인지를 유추할 수 있도록 원데이터 수를 증가시켜 가면서 SOM을 학습시키고 quantization error를 측정해 보았다. 100개를 시작으로 50개씩 늘려가며 400개까지 적용하고 100개 일 때를 기준으로 하여 얼마만큼의 에러가 감소하는지 계산하였다. 데이터 개수 별로 50회 반복시행의 평균을 이용한 결과이다.

표 3. raw데이터 수에 따른 SOM의 학습에러-3gaus

raw data 수	100	150	200	250	300	350	400
error	0.2781	0.2727	0.2713	0.2643	0.2599	0.2567	0.2487
error감소율(%) (100개 기준)		1.94	2.45	4.96	6.54	7.70	10.57

표 4. raw데이터 수에 따른 SOM의 학습에러-Abalone

raw data 수	100	150	200	250	300	350	400
error	0.0906	0.0885	0.0859	0.084	0.0828	0.0816	0.0803
error감소율(%) (100개 기준)		2.32	5.19	7.28	8.61	9.93	11.37

표 5. raw데이터 수에 따른 SOM의 학습에러-Pima

raw data 수	100	150	200	250	300	350	400
error	0.1809	0.1779	0.1744	0.1705	0.1661	0.1654	0.1626
error감소율(%) (100개 기준)		1.66	3.59	5.75	8.18	8.57	10.12

먼저 3gaus data에 대해 살펴보면 알고리즘을 통해 평균적으로 2%정도의 에러 감소 효과를 볼 수 있음을 그림4를 통해 알 수 있는데 이는 원데이터가 100개 에서 150~200개 정도로 증가했을 때의 효과에 버금가는 수치다. 또한 이 데이터의 경우 최고 6%정도의 에러감소를 얻을 수 있는 것으로 관찰되고 있는데 이것은 원데이터가 250개 이상으로 증가했을 때의 수치에 해당한다.

다음으로 Abalone 데이터를 보면 평균적으로 4%정도의 감소효과를 기대할 수 있으며 이는 원데이터가 100개에서 200개 정도로 증가할 때의 효과와 비슷하다. 또한 최대 감소율을 8% 내외로 본다면 이것은 원데이터가 250개가 되었을 때와 비슷하다.

마지막으로 Pima_indian 데이터를 살펴보면 평균 에러감소율이 1.5%정도로 원데이터가 150개 정도로 증가했을 때에 해당하는 수치다. 얻을 수 있는 최대 에러감소율을 4.5% 정도로 본다면 이는 원데이터 200개 정도 되었을 때에 해당한다.

5.2 full set에 대한 비중에 따른 효과

원데이터의 full set 데이터에 대한 비중에 따라 얻을 수 있는 효과 정도를 알아본다. 원데이터를 100, 200, 300, 400, 500 개로 고정시키고 알고리즘을 적용하여 에러 감소 효과가 어떻게 나타나는지 실험해 보았다. 루프횟수는 앞 절의 결과에서 가장 안정적이었던 2회로 고정하였다. 데이터 개수 별로 50회 반복실험 하여 평균, 최대, 최소 에러 감소율을 산출하였다.

표 6. 시작데이터 수에 따른 성과-3gaus

원데이터수	100	200	300	400	500
Average	2.128	1.918	1.157	0.573	0.21
Max	4.63	4.29	3.55	2.057	1.98
Min	-5.08	-5.71	-5.927	-6.533	-8.549
성공률	46/50	41/50	31/50	23/50	15/50

표 7. 시작데이터 수에 따른 성과-Abalone

원데이터수	100	200	300	400	500
Average	3.35	3.23	2.38	1.26	0.972
Max	6.18	5.79	4.25	3.54	2.5712
Min	-0.275	-1.58	-2.14	-2.55	-4.92
성공률	47/50	44/50	37/50	28/50	26/50

표 8. 시작데이터 수에 따른 성과-Pima

원데이터수	100	200	300	400	500
Average	1.676	1.7687	1.1677	0.9067	0.8973
Max	4.0464	3.5282	4.1815	2.6823	3.0513
Min	-0.405	-0.2942	-1.4743	-1.3249	-1.0311
성공률	47/50	45/50	38/50	32/50	25/50

원데이터가 많아질수록, 즉 full set 데이터에 대한 비중을 높일수록 효과가 감소하고 있음을 알 수 있다. 100% 모두 사용했을 경우엔 성공률도 50% 이하로 떨어지는데 이것은 당연한

결과다. 학습데이터와 테스트데이터가 동일한 상황에서 인공데이터가 첨가될 경우 이는 노이즈로 작용할 가능성이 크기 때문이다. 에러가 향상되는 경우가 있는 것은 SOM의 iterative한 학습과정이 optimal한 해를 찾지 못한다는 사실과 부합되는 것으로 볼 수 있다.

결과를 종합해 보면 이 알고리즘은 학습데이터가 데이터모집단의 절반 이하로 주어질 때 최적의 성능을 보일 수 있을 것으로 생각된다.

5.3 알고리즘 일반성능

마지막으로 다양한 데이터에 대해 알고리즘을 적용하여 일반성능을 알아본다.

사용된 데이터는 총 7가지 이며 세부 정보는 아래 표와 같다. 이 데이터들은 모두 UCI machine learning repository에 포함된 것들이다. 원데이터는 100개, 루프횟수는 2회로 고정하고 데이터마다 50회의 반복시행 후 최대, 최소 감소율, 평균 감소율을 산출하였다.

표 9. 데이터셋 설명

Data	개수	차원
Glass	214	9
Housing	506	12
Image	209	16
Liver	345	6
Wine	178	13
Yeast	1484	7
Iris	150	3

결과는 다음 표와 같다.

표 10. 결과

Data	성공율(%)	Min	Max	Average
Glass	48/50	-1.27	8.38	4.70
Housing	47/50	-0.64	9.40	3.17
Image	48/50	-0.20	8.99	3.76
Liver	42/50	-1.09	3.76	1.36
Wine	49/50	-0.50	3.47	2.11
Yeast	39/50	-1.06	1.57	0.62
Iris	47/50	-0.31	7.58	3.35

일반 데이터에 대한 실험에서 대부분의 데이터에 대해 알고리즘이 좋은 효과를 내고 있다. 단, Yeast데이터에 대해서는 그다지 효과를 거둘 수 없었는데 앞서 실험에 사용한 3gaus데이터와 더불어 좋은 성과가 나지 않는 데이터들에 대한 부가적인 연구를 통해 이들 데이터의 특성을 파악할 필요할 것으로 생각된다.

6. 결론

지금까지 SOM의 학습과정에 인공의 데이터를 첨가하여 quantization error를 줄이는 알고리즘을 소개 하였다. 다양한 데이터에 대한 실험에서 제안된 알고리즘이 부족한 학습데이터를 보완하여 SOM의 학습에 도움을 줄 수 있을 것으로 나타났다. 그러나 몇몇 데이터에서 알고리즘이 좋은 효과를 낼 수 없었다는 점은 한계점으로 남게 되었다. 앞으로 이러한 데이터에 대한 연구를 통해 그 특성을 밝혀내고 알고리즘을 보완할 필요가 있을 것으로 생각된다.

References

[1] T. Kohonen, Self-Organization and Associative Memory, Springer Series in Information Sciences, Springer, New York, 1988.

[2] T. Kohonen, "The self organizing map", proceedings of the IEEE, Vol.78 No.9(1990) pp.1464-1480.

[3] H. J. Ritter, T. M. Martinetz, and K. J. Schulten. Neuronale Netze. Addison-Wesley, Munchen, 1991.

[4] M. Jang, S. Cho, Ensemble learning algorithm using virtual data,

[5] G.An, The effects of adding noise during backpropagation training on a generalization performance, Neural Computation, 7(2):613-674, 1996.

[6] Yuan Jiang, Zhi hua Zhou, SOM ensemble based image segmentation, Neural Processing Letters 20:171-178, 2004

[7] S.Cho, Ensemble of structure-adaptive self-organizing maps for high performance classification, Information Sciences, Volume 123, Issues 1-2 : 103-114, 2000