Invited Talk, Proc. of World Congress on Neural Networks (WCNN95), July 17-21, 1995, Washington, DC, Vol.I, pp.35-42.

A Unified Learning Framework: Multisets Modeling Learning[1]

Lei Xu
1. Dept. of Computer Science, The Chinese University of Hong Kong
Shatin, Hong Kong (the correspondence address)
2. Information Science Center, Peking University, Beijing, China

Abstract *A unified learning framework is proposed. Its different special cases will automatically lead us to current existing major types of neural network learnings, e.g, data clustering, various PCA-type self-organizations and their localized extensions, self-organizing topological map, as well as supervised learning for feedforward network and modular architecture. Not only this new framework is useful for a deep understanding of the existing learnings, but also it provides new insights that can guide the extensions of the existing learning methods. With the new insights, we have obtained several new intersting unsupervised learnings, and introduced regularization or generalization to unsupervised learnings, particularly proposed approaches for solving a classical hard problem—how to decide the number of clusters in clustering analysis or competitive learning.*

1. Introduction

Unsupervised and supervised learnings are regarded being essentially different and have been studied separately. Unsupervised learning alone has even three major types: clustering, *Principal Component Analysis (PCA)*-type self-organizing, and topological map. They are developed from different motivations and based on different theories or heuristics, and considered to be basically different from each other.

Multisets Modeling Learning (MML) proposed in Xu (1994) intends to unify all these learnings into one single framework. The key idea is to use a general real set specified by a given property as the representation of a model and a number of such sets as the joint representation of multiple models. The purpose of learning is to determine each model set, which is specified by some parametric equation, to minimize the error of this model on approximating a data set. Using the framework, we can naturally unify various unsupervised learning such as *clustering*, standard *PCA*, *local PCA*, *Minor Component Analysis (MCA)*, *local MCA*, *k-PCA*, *Principal Subspace Analysis (PSA)* , *k-MCA*, *Minor Subspace Analysis (MSA)* and their localized extensions. This paper presents further development on the MML framework such that self-organizing topological map (Kohonen, 1989) and supervised learning on feedforward networks and on modular architecture of local experts (Jacobs, Jordan, Nowlan, & Hinton , 1991) can also naturally unified. With the new framework, we have also obtained several new intersting unsupervised learnings, introduced regularization or generalization to unsupervised learnings, and proposed the approaches for how to decide the number of clusters in clustering analysis or competitive learning. We have also shown that the framework can be related to Maximum likelihood learning of finite non-gaussian mixtures.

2. The General Framework of MML

Given a finite or continuous set S of real points in R^d, we represent a neural network model by a set:

$$M = \{x : p(x) \ and \ x \in S\}, \tag{1}$$

where $p(x)$ means that x satisfies the properties specified by a general predicate proposition p. For examples, $p(x)$ may mean that x is a root of an equation $F(x) = 0$, or that $x = [\xi, \eta]$ satisfies an explicit function $\eta = f(\xi)$, as well as that x makes $F(x) > 0$ or $F(x) < 0$. Moreover, $p(x)$ can also be a combined proposition that consists of a number of such simple propositions $p_1(x), \cdots, p_r(x)$ via logic connectives $\wedge, \vee, \neg$. As a result, a model M can be a point, a curve, a surface, an area or volume with any shape or any of their combinations and it is either able or unable to be described by mathematical equations. Moreover, it even can be described by languages. In summary, *this model can be any set in* R^d.

To use such M to model a given data set, we define that M represents x_i with an error given by

$$\varepsilon^q(x_i, M) = \min_{y \in M \cap R} |T(x_i - y)|^q, \ q \geq 1, \tag{2}$$

where $|u|^q = \sum_{i=1}^n |u_i|^q$ for $u = [u_1, \cdots, u_n]^T$, and T is an $d \times d$ nonsingular matrix. We have $\varepsilon^q(x_i, M) = 0$ when $x_i \in M \cap R$. $R \subseteq M$ acts as some constraint on y. One common case is $R = M$, in this case $y \in M \cap R$ becomes $y \in M$ and there is no constraint. The other case is that

$$R = R(x_i) = \{x : \xi = \xi_i, x = [\xi, \eta], x_i = [\xi_i, \eta_i], \ x \in M\} \tag{3}$$

with ξ, ξ_i consisting of k variables of x, x_i respectively $(0 < k < d)$. When $q = 2$, $|T(x_i - y)|^2 = (x_i - y)^t\Sigma(x_i - y)$ is Mahalanobis distance since $\Sigma = T^tT$ is positively defined. The function of T is a linear

[1]This project was supported in part by the HK RGC Earmarked Grant CUHK250/94E.

transformation of coordinate system. When $T = I$, $\|T(x_i - y)\|^2 = \|x_i - y\|^2$ is the square distance between x_i, y. When $R = M$, there is no extra constraint. In this case, eq.(2) becomes

$$\varepsilon^2(x_i, M) = \min_{y \in M} \|x_i - y\|^2, \tag{4}$$

The followings are some interesting special cases:

$$\begin{aligned}
\varepsilon^2(x_i, M_a) &= \|x_i - a\|^2, \quad M = M_a = \{a\}; \\
\varepsilon^2(x_i, M_L) &= \|[I - (w-a)(w-a)^t](x-a)\|^2, \quad M = M_L = \{x : x - a \textit{ parallels } w - a\}; \\
\varepsilon^2(x_i, M_P) &= \frac{[(w-a)^t(x_i - a)]^2}{\|w-a\|^2}, \quad M = M_P = \{x : (x-a)^t(w-a) = 0\}; \\
\varepsilon^2(x_i, M_M) &= \|(I-P)(x-a)\|^2, \quad P = W(W^tW)^{-1}W^t. \quad M_M = \{y : y = x + a, x \in S\}; \\
W &= [w_1 - a, \cdots, w_k - a], \textit{ subspace } S \textit{ spanned by independent vectors } w_1 - a, \cdots, w_k - a; \\
\varepsilon^2(x_i, M_S) &= |c^2 - \|x-a\|^2|, \quad M = M_S = \{x : \|x-a\|^2 = c^2\}.
\end{aligned} \tag{5}$$

M_L is a *line* passing through a point a, M_P is a *hyperplane* passing through a point a, and M_M is a *linear manifold* — a shifted subspace that locates at a point a. Moreover, M_S is a *sphere* of radius c that locates at a. The corresponding error $\varepsilon^2(x_i, M)$ is actually the shortest distance of the point x_i to the line, hyperplane, linear manifold and sphere respectively.

More generally, we can modify eq.(2) by adding a regularization term $r(M) > 0$ to penalize the complexity of M:

$$e^q(x_i, M) = \beta_r r(M) + \varepsilon^q(x_i) = \beta_r r(M) + \min_{y \in M \cap R} |T(x_i - y)|^q, \; q \geq 1, \; \beta_r > 0. \tag{6}$$

Given a data set $\mathcal{D}_x = \{x_1, \cdots, x_N\}$, the aim of learning is to specify M such that $\sum_{i=1}^N e^q(x_i)$ is minimized, we concentrate on cases that M can be determined by a set of parameters Ψ, e.g., $\Psi = \{a\}$ for M_a, $\Psi = \{a, w\}$ for M_L and M_P, $\Psi = \{W\} = \{a_1, \cdots, a_k, w_1, \cdots, w_k\}$ for M_M. Our aim is to minimize the following J^q with respect to Ψ:

$$J^q = \sum_{i=1}^N e^q(x_i, \Psi) = \sum_{i=1}^N [\beta_r r(\Psi) + \varepsilon^q(x_i, \Psi)]. \tag{7}$$

Further generally, for a multi-modes data $\mathcal{D}_x$ that comes from a mixture of several different objects or models. We propose to use a number of models $M_m(\Psi_m), m = 1, \cdots, N_c$ to deal with these cases. The learning problem is now specified by the minimization of the mixture error with respective to $\Psi_1, \cdots, \Psi_{N_c}$:

$$J^q_{N_c} = \sum_{m=1}^{N_c} J^q_m, \quad J^q_m = \sum_{x \in S_m} [\beta_r r(\Psi_m) + \varepsilon^q(x_i, \Psi_m)], \quad S_m = \{x_i : e^q(x_i, \Psi_m) < e^q(x_i, \Psi_j), \; j \neq m\} \tag{8}$$

The direct minimization $J^q_{N_c}$ is a combinatorial problem of partitioning $\mathcal{D}_x$ into subsets $\{S_m\}_1^{N_c}$ so that $J^q_{N_c}$ reaches its minimum. A *Hard Cut Iterative (HCI) Algorithm* can be proposed for this minimization. Initially, we divide $\mathcal{D}_x$ into exclusive subsets $\{S_m\}_1^{N_c}$, and then repeat the two steps to find a local minimum of $J^q_{N_c}$:

Step 1 Take $x \in \mathcal{D}_x$, assume it is currently in S_i, we check the possibel changes $\Delta_i = -\Delta J^q_i$ and $\Delta_j = \Delta J^q_j, \;\; j \neq i$ that may result in if we remove it from S_i to S_j for $j \neq i$. The changes can be calculated according to eq.(8).

Step 2 Find S_r with $\Delta_r = \min_j \Delta_j$, put x into S_r, then, goto Step 1. The iteration stops until no change for all $x \in \mathcal{D}_x$.

An alternative for eq.(8) is to soften the hard cut partition by a set of auxiliary variables $\{\omega_{mi}, m = 1, \cdots, N_c; i = 1, \cdots, N\}$ and changing eq.(8) into

$$J^q_{N_c} = \sum_{m=1}^{N_c} \sum_{i=1}^N \omega_{mi}[\beta_r r(\Psi_m) + \varepsilon^q(x_i, \Psi_m)] + \beta \sum_{m=1}^{N_c} \sum_{i=1}^N \omega_{mi} \ln \omega_{mi}, \; \sum_{m=1}^{N_c} \omega_{mi} = 1. \tag{9}$$

Where $1 \geq \omega_{mi} \geq 0$ denotes the probability of x_i generated from $M_m(\Psi_m)$. ω_{mi}'s are unknown and need to be determined through the above minimization. The above 2nd term is used for avoiding the problem of over-sized unknowns. The term can be interpreted as the negative entropy of distribution ω_{mi}. It means that we try to keep those unknowns instead of imposing some additional constraints. β is a given weight variable which controls the portion of the 2nd term in the whole cost.

The minimization eq.(9) can be implemented by two iterative steps of the following *Alternative Optimization (AO)* algorithm :

Step 1 With the parameters $\Psi_m^{(k)}, m = 1, \cdots, M$ fixed, considering the constraint $\sum_{m=1}^{N_c} \omega_{mi} = 1$ and by $\nabla_{\omega_{mi}} J^q_{N_c} = 0$, we can get

$$\omega_{mi} = e^{-\beta^{-1}[\varepsilon^q(x_i, \Psi_m^{(k)}) + \beta_r r(\Psi_m)]} / \sum_{m=1}^{N_c} e^{-\beta^{-1}[\varepsilon^q(x_i, \Psi_m^{(k)}) + \beta_r r(\Psi_m)]}. \tag{10}$$

Step 2 With the parameters ω_{mi}'s fixed, the minimization of J^q_M with respect to $\Psi_m, m = 1, \cdots, M$ can be decomposed into each of the independent minimizations with respect to Ψ_m

$$J^q_{M_m}(\Psi_m) = \sum_{i=1}^{N} \omega_{mi}[\varepsilon^q(x_i, \Psi_m) + \beta_r r(\Psi_m)], \tag{11}$$

which is just the weighted version of eq.(7) and can be solved in the same way as in single model, resulting in the updated $\Psi_m^{(k+1)}, m = 1, \cdots, N_c$.

Given the initial $\Psi_m^{(0)}, m = 1, \cdots, N_c$, we can repeatedly implement the above two steps. The iteration will converge to at least a local minimum of J^q_M as long as the second step let $J^q_{M_m}(\Psi_m^{(k+1)}) \le J^q_{M_m}(\Psi_m^{(k)}), m = 1, \cdots, N_c$ with $J^q_{M_m}(\Psi_m^{(k+1)}) \ne J^q_{M_m}(\Psi_m^{(k)})$ for a number of iterations. The reason is that each step is actually doing a descent search of $J^q_{N_c}$.

3. Unifying the Existing Major Types of Learnings

We show that the different special cases will automatically led to existing major types of learnings.

3.1 Unsupervised Learning: Single Model

We first consider the case of single model, i.e., $N_c = 1$. We consider the special case that $\beta_r = 0$, $T = I$, $R = M$ and $q = 2$ in eqs.(6)(7), i.e., the unconstrained square error without the regularization term $r(\Psi_m)$.

For the special cases given in eq.(5), the cost eq.(7) becomes

$$\begin{aligned}
J^2_a &= \sum_{i=1}^{N} \|x_i - a\|^2, \quad for\ M = M_a \\
J^2_L &= \sum_{i=1}^{N} \|[I - (w-a)(w-a)^t](x-a)\|^2, \quad for\ M = M_L \\
J^2_P &= \sum_{i=1}^{N} \frac{[(w-a)^t(x_i - a)]^2}{\|w-a\|^2}, \quad for\ M = M_P \\
J^2_M &= \sum_{i=1}^{N} \|(I-P)(x-a)\|^2, \quad P = W(W^tW)^{-1}W^t, \quad for\ M = M_M \\
J^2_{M1} &= \sum_{i=1}^{N} \|(I - WW^T)(x-a)\|^2, \quad for\ M = M_M\ and\ P = WW^T, \\
J^2_{M2} &= \sum_{i=1}^{N} \|(I - WW^T)(x-a)\|^2_{W^TW=I}, \quad for\ M = M_M,\ P = WW^T,\ W^TW = I.
\end{aligned} \tag{12}$$

These minimizations with respect to a give the mean vector $a = \frac{1}{N}\sum_{i=1}^{N} x_i$, which represents the learned location of data $\mathcal{D}_x$. For M_a, this is a simplest unsupervised learning task. As to the minimization with respect to w or W, we have

(1) **PCA** For a line M_L, $w - a$ is the eigenvector of $\Sigma = \frac{1}{N}\sum_{i=1}^{N}(x_i - a)(x_i - a)^T$ that corresponds to the largest eigenvalue. That is, the minimization led to PCA (Oja, 1982,89; Xu, 1991, 93, 94). When $a = 0$, J^2_L also becomes a special case of the LMSER rule proposed in (Xu, 1991,93) for a single linear neuron.

(2) **MCA** For a hyperplane M_L, $w - a$ is the eigenvector of Σ that corresponds to the smallest eigenvalue. That is, the minimization led to MCA, which has been used for curve fitting (Xu & Oja, 1992).

(3) **k-PCA, PSA** For a linear manifold M_M that corresponds to J^2_M, $J^2_{M_1}$ and $J^2_{M_2}$, we have the solution $W = \Phi R$, Φ consists of the eigenvectors corresponding to the k largest eigenvalues of Σ as its column vectors, and $R^t R = I$. That is, the learning performs PSA (Oja , 1989; Xu, 1991, 93, 94). Particularly, for $J^2_{M_2}$ we will have $R = I$ and $W = \Phi$, i.e, the learning finds the first k principal components (k-PCA). Moreover, $J^2_{M_1}$ with $a = 0$ becomes the special case of the LMSER proposed in (Xu, 1991, 93) for one layer linear network that performs PSA.

(4) **k-MCA, MSA** If we replace the subspace S in M_M by its orthogonal complement subspace $\bar{S}$, then the above learning will performs k-MCA or MSA.

3.2. Unsupervised Learning: Multiple Models

We consider the case of multiple model with $N_c > 1$. Now we could not get analytical solution, and the solutions are obtained through the iterative algorithm HIC or AO. Again, we fix $\beta_r = 0$, $T = I$, $R = M$ and $q = 2$ in eqs.(8)(9).

Clustering or Vector Quantization (VQ) For the special case that each M_i is a point a_i, eq.(8) becomes the minimization of

$$J^2_{N_c} = \sum_{m=1}^{N_c} J^q_m, \quad J^q_m = \sum_{x \in S_m} \|x_i - a_m\|^2, \quad S_m = \{x_i : \|x_i - a_m\| < \|x_i - a_j\|,\ j \neq m\} \tag{13}$$

which is the classical *Clustering* or in other words, *Vector Quantization (VQ)*. It is not difficult to see that the HCI algorithm returns to the conventional k-mean algorithm.

The learning problem eq.(9) will become the minimization of

$$J^2_{N_c} = \sum_{m=1}^{N_c} \sum_{i=1}^{N} \omega_{mi} \|x_i - a_m\|^2 + \beta \sum_{m=1}^{N_c} \sum_{i=1}^{N} \omega_{mi} \ln \omega_{mi}, \quad s.t. \sum_{m=1}^{N_c} \omega_{mi} = 1 \tag{14}$$

For the algorithm AO, the two steps eq.(10) and eq.(11) will become

$$\omega_{mi} = e^{-\beta^{-1}\|x_i - a_m^{(k)}\|^2} / \sum_{m=1}^{N_c} e^{-\beta^{-1}\|x_i - a_m^{(k)}\|^2}, \quad a_m^{(k)} = \frac{1}{n_m} \sum_{i=1}^{N} \omega_{mi} x_i, \quad n_m = \sum_{i=1}^{N} \omega_{mi}. \tag{15}$$

This is identical to the EM algorithm for Gaussian mixtures with equal prior probabilities and given equal covariance (Xu & Jordan, 1993b), or can be regarded as a soft k-means algorithm.

Local LMSER or Local PCA For the special case that each M_i is a line M_L given in eq.(5), we have that eq.(8) becomes the minimization of

$$J^q_{N_c} = \sum_{m=1}^{N_c} J^q_m, \quad J^q_m = \sum_{x \in S_m} \|[I - (w_m - a_m)(w_m - a_m)^t](x - a_m)\|^2$$

$$S_m = \{x_i : \|[I - (w_m - a_m)(w_m - a_m)^t](x - a_m)\| < \|[I - (w_j - a_j)(w_j - a_j)^t](x - a_j)\|,\ j \neq m\} \tag{16}$$

which becomes the Local Least MSE Reconstruction (LLMSER) learning proposed in Xu(1995). The result is that each of $a_1, \cdots, a_{N_c}$ becomes a cluster center and each of $(w_1 - a_1), \cdots, (w_{N_c} - a_{N_c})$ becomes the principal component of the corresponding cluster. That is, it performs *local PCA* (Xu, 1994; Leen & Kambhatla, 1993). The HCI algorithm becomes the *Hard Cut Local LMSER-VQ* algorithm given in Xu(1995).

The learning problem eq.(9) will become the minimization of

$$J^2_{N_c} = \sum_{m=1}^{N_c} \sum_{i=1}^{N} \omega_{mi} \|[I - (w_m - a_m)(w_m - a_m)^t](x_i - a_m)\|^2 + \beta \sum_{m=1}^{N_c} \sum_{i=1}^{N} \omega_{mi} \ln \omega_{mi}, \quad s.t. \sum_{m=1}^{N_c} \omega_{mi} = 1 \tag{17}$$

For the algorithm AO, eq.(10) becomes

$$\omega_{mi} = e^{-\beta^{-1}\|[I-(w_m-a_m)(w_m-a_m)^t](x_i - a_m^{(k)})\|^2} / \sum_{m=1}^{N_c} e^{-\beta^{-1}\|[I-(w_m-a_m)(w_m-a_m)^t](x_i - a_m^{(k)})\|^2} \tag{18}$$

and eq.(11) can be explicitly solved such that $a_m^{(k)} = \frac{1}{n_m}\sum_{i=1}^N \omega_{mi}x_i$, $n_m = \sum_{i=1}^N \omega_{mi}$ and $(w_m^{(k)} - a_m^{(k)})$ is the eigenvector of

$$\Sigma_m^{(k)} = \frac{1}{n_m}\sum_{i=1}^N \omega_{mi}(x_i - a_m^{(k)})(x_i - a_m^{(k)})^T, m = 1, \cdots, N. \tag{19}$$

that corresponds to the largest eigenvalue. This algorithm is actually the *Soft Local LMSER-VQ II* proposed in Xu(1995).

3.3 Supervised Learning

Feedforward Net We consider the single model eq.(2) with $R = R(x_i)$ given in eq.(3). Also we let $T = I$.

For the case that M is specified by an explicit function $\eta = f(\xi, \Psi)$, we partition y in eq.(2) into $y = [\xi, \eta]$ according to the training set $\mathcal{D}_x = \mathcal{D}_{[\xi,\eta]} = \{[\xi_1, \eta_1], \cdots, [\xi_N, \eta_N]\}$. Under the constraint eq.(3), the cost eq.(7) becomes

$$J^q = \sum_{i=1}^N \{|[\eta_i - f(\xi_i, \Psi)|^q + \beta_r r(\Psi)\}, \tag{20}$$

with $f(\xi_i, \Psi)$ represented by a feedforward network and Ψ being it parameters. The problem of minimizing this J^q or particularly J^2 to decide Ψ is just the usual supervised learning by the least L_p error or square error with a regularization term $r(\Psi)$, which can be done by *backpropagation* method.

Mixtures of Experts For the cases that multi-sets $M_m, m = 1, \cdots, N_c$ are specified by explicit functions $f_m(\xi, \Psi_m), m = 1, \cdots, N_c$, eq.(9) becomes the problem of minimizing

$$J^q_{N_c} = \sum_{m=1}^{N_c} J^q_m, \quad J^q_m = |\eta_i - f_k(\xi_i, \Psi_m)|^q + \beta_r r(\Psi_m), \quad S_m = \{[\eta_i, \xi_i] : J^q_m < J^q_j, \ j \neq m\} \tag{21}$$

This is a modularized supervised learning. In general, its solution by the HCI algorithm is difficult when $f_j, j = 1, \cdots N_c$ are nonlinear functions.

Alternatively, the learning problem eq.(9) will become the minimization

$$J^q_{N_c} = \sum_{m=1}^{N_c}\sum_{i=1}^N \omega_{mi}[|\eta_i - f_m(\xi_i, \Psi_m)|^q + \beta_r r(\Psi_m)] + \beta \sum_{m=1}^{N_c}\sum_{i=1}^N \omega_{mi}\ln\omega_{mi}, \ s.t. \sum_{m=1}^{N_c}\omega_{mi} = 1, \tag{22}$$

For the algorithm AO, eq.(10) and eq.(11) becomes

$$\begin{aligned} \omega_{mi} &= e^{-\beta^{-1}[|\eta_i - f_m(\xi_i, \Psi_m)|^q + \beta_r r(\Psi_m)]} / \sum_{j=1}^{N_c} e^{-\beta^{-1}[|\eta_i - f_j(\xi_i, \Psi_j)|^q + \beta_r r(\Psi_m)]} \\ J^q_{M_m}(\Psi_m) &= \sum_{i=1}^N \omega_{mi}[|\eta_i - f_m(\xi_i, \Psi_m)|^q + \beta_r r(\Psi_m)], \end{aligned} \tag{23}$$

When $q = 2$ and $\beta_r = 0$ (i.e., no the regularization term). this supervised learning framework eq.(22) and the AO algorithm are actually the *mixtures of experts* and its EM learning algorithm (Jordan & Jacobs, 1994; Xu, Jordan & Hinton, 1994).

3.4 Self-Organizing Topological Maps

We reconsider the special case that each M_i is a point a_i, with the regularization term added in eq.(14),

$$J^2_{N_c} = \sum_{m=1}^{N_c}\sum_{i=1}^N \omega_{mi}[\|x_i - a_m\|^2 + \beta_r r(a_m)] + \beta \sum_{m=1}^{N_c}\sum_{i=1}^N \omega_{mi}\ln\omega_{mi}, \quad s.t. \ \sum_{m=1}^{N_c}\omega_{mi} = 1 \tag{24}$$

We re-arrange $a_1, \cdots, a_{N_c}$ on a lattice structure $\{a_{i,j}, i = 1, \cdots, N_c^{(1)}, j = 1, \cdots, N_c^{(2)}\}$, $N_c^{(1)}N_c^{(2)} = N_c$, and re-arrange eq.(24) into

$$\begin{aligned} J^2_{N_c} &= \sum_{i=1}^{N_c^{(1)}}\sum_{j=1}^{N_c^{(2)}}\sum_{t=1}^N \omega_{i,j,t}[\|x_i - a_{i,j}\|^2 + \beta_r r(a_{i,j})] + \beta \sum_{i=1}^{N_c^{(1)}}\sum_{j=1}^{N_c^{(2)}}\sum_{i=1}^N \omega_{i,j,t}\ln\omega_{i,j,t}, \quad s.t. \ \sum_{i=1}^{N_c^{(1)}}\sum_{j=1}^{N_c^{(2)}} \omega_{i,j,t} = 1, \\ r(a_{i,j}) &= \sum_{p,q \in N_{(i,j)}} \|a_{p,q} - a_{i,j}\|^2. \end{aligned} \tag{25}$$

where $N_{(i,j)}$ is a pre-specified neighbor set of the unit (i,j), for example, for the 4-neighbors of (i,j) we have $N_{(i,j)} = \{(i,j-1),(i,j+1),(i,j),(i-1,j),(i+1,j)\}$ and $r(a_{i,j})$ becomes

$$r(a_{i,j}) = \|a_{i,j} - a_{i,j-1}\|^2 + \|a_{i,j} - a_{i,j+1}\|^2 + \|a_{i-1,j} - a_{i,j}\|^2 + \|a_{i+1,j} - a_{i,j}\|^2 \tag{26}$$

We minimize this $J^2_{N_c}$ by the algorithm AO, with eq.(10) becoming

$$\omega_{i,j,t} = e^{-\beta^{-1}[\|x_i - a^{(k)}_{i,j}\|^2 + \beta_r r(a_{i,j})]} / \sum_{i=1}^{N_c^{(1)}} \sum_{j=1}^{N_c^{(2)}} e^{-\beta^{-1}[\|x_i - a^{(k)}_{i,j}\|^2 + \beta_r r(a_{i,j})]} \tag{27}$$

eq.(11) becoming

$$\min_{\{a_{i,j}\}} \sum_{t=1}^{N} \omega_{i,j,t}[\|x_t - a_{i,j}\|^2 + \beta_r r(\{a_{i,j}\})] \tag{28}$$

The solution of this algorithm will be a topological map, similar to Kohonen map (Kohonen, 1989).

Actually, via several steps of approximate simplifications, we can obtain Kohonen map algorithm from the above algorithm. First, we let $\beta \to 0$, all the $\omega_{i,j,t}$'s above will tend to 0 except that one tends to 1, i.e., one becomes the winner. Suppose that the winner is $\omega_{i^*,j^*,t}$. Second, we consider $\min_{a_{i^*,j^*}} \|x_t - a_{i^*,j^*}\|^2 + \beta_r r(a_{i^*,j^*})$ by stochastic approximation. we update a_{i^*,j^*} by gradient descent $\Delta a_{i^*,j^*} = -\alpha(x_t - a_{i^*,j^*})$ through neglecting the 2nd term in eq.(28). Third, we minimize this 2nd term by letting the neighbors of a_{i^*,j^*} to approach a_{i^*,j^*}, which gives that $a_{p,q} = a_{i^*,j^*}$ for $p,q \in N_{(i^*,j^*)}$. In summary, we have the following stochastic approximation algorithm—Kohonen map algorithm:

$$\Delta a_{i,j} = \begin{cases} -\alpha(x_t - a_{i,j}), & \text{if } (i,j) \in N_{(i^*,j^*)}, \\ 0, & \text{otherwise.} \end{cases} \tag{29}$$

It is also no difficulty to re-arrange $a_1, \cdots, a_{N_c}$ on an one dimensional or high dimensional structure to self-organize the corresponding topological structures.

4. Providing New Results and Research Topics for Learning

This framework provides a unified understanding on various existing neural network learning. Although they may look quite different, the deep basic mechanism behind them is the same at the level of modeling by sets. Their surficial differences are due to the different detail structures and constraints on each model set. Moreover, as will be shown below, this framework also provides new results and new research topics for learning studies.

4.1 Local Unsupervised Learnings By Sets With Increasing Complexities

In sec. 3.2, we have shown that *clustering* is actually a localized modeling with points and that *Local PCA or LMSER* is actually a localized modeling with lines. Similarly, we can increase the complexity of set M_m to the hyperplane M_P, the linear manifold M_M, the sphere M_S as well as many other sets with increasing complexities. Along this direction, many interesting new results and research topics can be obtained. Some examples are given below.

Local PCA and Multi-line Total Least Square Fitting For the special case that each M_i is a hyperplane M_P given in eq.(5), we have that eq.(8) and eq.(9) become the minimizations of

$$J^q_{N_c} = \sum_{m=1}^{N_c} J^q_m, \quad J^q_m = \sum_{x \in S_m} \frac{[(w_m - a_m)^t (x - a_m)]^2}{(w_m - a_m)^t (w_m - a_m)}$$

$$S_m = \{x_i : \frac{[(w_m - a_m)^t (x - a_m)]^2}{(w_m - a_m)^t (w_m - a_m)} < \frac{[(w_j - a_j)^t (x - a_j)]^2}{(w_j - a_j)^t (w_j - a_j)}, \ j \neq m\} \tag{30}$$

$$J^2_{N_c} = \sum_{m=1}^{N_c} \sum_{i=1}^{N} \omega_{mi} \frac{[(w_m - a_m)^t (x - a_m)]^2}{(w_m - a_m)^t (w_m - a_m)} + \beta \sum_{m=1}^{N_c} \sum_{i=1}^{N} \omega_{mi} \ln \omega_{mi}, \quad s.t. \sum_{m=1}^{N_c} \omega_{mi} = 1 \tag{31}$$

Eq(30) and eq(31) can be solved by the algorithms HCI and AO respectively, which result in a_m as the local cluster center, and $(w_m - a_m)$ as the minor component direction of this cluster [see Xu (1994) for detail]. That is, this learning performs local MCA. Actually, this provides a new approach for detecting and fitting a number of lines or hyperplances in the sense of the total least square error. The approach extends those methods proposed in Xu, Oja & Suen (1992) and Xu, Krzyzak & Oja (1993), and it will be very useful for line and plane detection for image recognition and computer vision.

Local Subspaces For the special case that each M_i is a linear manifold M_M that corresponds to J^2_M, $J^2_{M_1}$ and $J^2_{M_2}$, and are given in eqs.(5) (12), then similarly by solving eq.(8) or eq.(9) via the algorithm HCI or AO, we can obtain each subspace located at each cluster's center a_m. That is, the learning performs *local k-PCA* or *local PSA*. Moreover, Replacing M_M, M_{M_1}, M_{M_1} by their corresponding orthogonal complements, we can also get *local k-MCA* or *local MSA*. These new results make it possible to extend the so called *Subspace Pattern Recognition and Dual Subspace Pattern Recognition* (Oja, 1983; Xu & Oja, 1991) from supervised learning based approaches to unsupervised learning based approaches, so that more practical pattern recognition problems can be attacked.

Multi-circles or spheres Total Least Square Fitting For the special case that each M_i is a sphere M_S given in eqs.(5) (12), by solving eq.(8) or eq.(9) via the algorithm HCI or AO, we can let each M_i to detect and represent one of spheres that locates at different points $a_1, \cdots, a_{N_c}$. This property can be applied to computer vision for robust detection of circle (Xu & Oja, 1993).

We can further increase the complexity of M_i to ellipse or any other shapes as well as ball, cubic and any volumes. Different types of sets can also be mixed together in the same modeling. As a result, we can get many interesting local unsupervised learning, such as *local surface fitting*, *local volumes fitting*, for tackling various density estimation, pattern recognition and computer vision problems.

4.2 Regularization or Generalization for Unsupervised Learnings

In the literature of neural network learning, the issues on regularization or generalization for supervised learning on feedforward networks have been investigated vastly, such as adding a regularization term, VC dimension and generalization error, MDL, Bayesian modeling.

However, few efforts have been maded on unsupervised learnings like *clustering*, *PCA* type learning and topological map. Here, the unification of supervised and unsupervised learning in a same framework makes it possible to extend the studies on regularization or generalization to various unsupervised learnings, which will give us many new results and interesting topics. In sequel, we discuss two simple but interesting cases.

The regularization term $r(\Psi) = tr(\Psi^t\Psi)$. This regularization has been widely used in training feedforward network for better generalization. How the cases discussed in sec.3 are influenced by adding such a term ?

For $M = M_a$, $r(\Psi) = \|a\|^2$, J^2_a given in eq.(12) will become $J^2_a + \beta_r\|a\|^2$ with solution $a' = \frac{N}{N+\beta_r}\frac{1}{N}\sum_{i=1}^N x_i$, i.e., it biases towards zero with its norm reduced by $\frac{N}{N+\beta_r}$. In the same time, the variance has reduced also by the factor $\frac{N}{N+\beta_r}$.

For the case of multi-sets with each $M_m = \{a_m\}$, apparently each center $a_m = \frac{1}{n_m}\sum_{i=1}^N \omega_{mi}$ should also contract by $\frac{n_m}{n_m+\beta_r}$. However, the change of a_m will also change ω_{mi}, which in turn will change a_m again. How the solution is actually affected ? It may shrink the spreading of the center's locations. It is an interesting topic for a further study. It is also interesting to investigate how the other cases discussed in sec.3 are affected by the term $r(\Psi) = tr(\Psi^t\Psi)$.

Minimum description length (MDL). For multi-sets case, such as clustering, PCA, local PCA, one of a key problem is how to select an appropriate N_c. The classical problem of selecting the number of clusters for a clustering algorithm (e.g., k-mean) is a typical example. With the unified framework, we know that the generalization approaches for supervised learning can also be used to unsupervised learnings. This motivates us to use MDL approach for this purpose.

Considering eq.(13), we examine the case of transmitting each data x_i from a sender to a receiver. Assume that $\{a_m\}_1^{N_c}$ are sent to the receiver in advance. For each data x_i, we first find an S_m such that $x_i \in S_m$, then we transmit two sets of coding bits to the receiver. One indicates which one of $\{a_m\}_1^{N_c}$ corresponds to $x_i \in S_m$, this needs $\ln N_c$ bits. The other are the bits for coding the residual $r_{im} = x_i - a_m$. Suppose r_{im} is from Gaussion $N(0, \sigma I)$, the coding bits of its j-th component are approximately $-\ln\{[\sqrt{2\pi}\sigma]^{-1}exp[-0.5(r^{(j)}_{im})^2/\sigma^2]\delta\} = 0.5(r^{(j)}_{im})^2/\sigma^2 + 0.5\ln(2\pi\sigma^2/\delta)$, where δ is quantization accuracy. The total bits for residuals of S_m are $0.5\sum_{x\in S_m}\|x_i - a_m\|^2/\sigma^2 + 0.5\#S_m\ln(2\pi\sigma^2/\delta)$. Therefore, the total bits for the first set is $N\ln N_c$ and the total bits for the second set are $0.5\sum_{m=1}^{N_c}\sum_{x\in S_m}\|x_i - a_m\|^2/\sigma^2 + 0.5N\ln(2\pi\sigma^2/\delta)$. Sum up them, we have that the MDL is equivalent to minimize

$$Jg^2_{N_c} = \frac{1}{N}\sum_{m=1}^{N_c}\sum_{x\in S_m}\|x_i - a_m\|^2 + 2\sigma^2\ln N_c + \sigma^2 ln(2\pi\sigma^2/\delta). \qquad (32)$$

The last term is independent of choosing N_c and determining $\{a_m\}_1^{N_c}$. By comparing with eq.(13), $Jg^2_{N_c}$ has the 2nd term which penalizes the large N_c. The larger the N_c is, the smaller the first term, but the larger the second term. This penalty also increases with the variance σ^2—the size of each cluster. The larger each cluster is supposed to be, the big the penalty is, the less the cluster's number. When σ^2 is known or can be specified, we can use eq.(32) to select a suitable N_c.

Eq.(32) can be further improved. In the first set of coding bits, each x_i uses $\ln N_c$ bits which implies that the probablity of assigning $x_i \in S_m$ is assumed to be $1/N_c$. But it is actually ω_{mi} given by eq.(15). So the first set of coding bits are $\sum_{m=1}^{N_c}\sum_{i=1}^{N}\omega_{mi}\ln\omega_{mi}$ instead of $N\ln N_c$. As a result, eq.(32) becomes

$$Jg^2_{N_c} = \frac{1}{N}\sum_{m=1}^{N_c}\sum_{x\in S_m}\|x_i - a_m\|^2 + \frac{2\sigma^2}{N}\sum_{m=1}^{N_c}\sum_{i=1}^{N}\omega_{mi}\ln\omega_{mi} + \sigma^2\ln(2\pi\sigma^2/\delta), \tag{33}$$

Again only the first two terms are related to the minimization with respect to N_c and $\{a_m\}_1^{N_c}$. Surprisingly, the first two terms form the $J^2_{N_c}$ in eq.(14) by regarding $\beta = 2\sigma^2$. In other words, our model eq.(14) for clustering is equivalent to do clustering based on MDL.

Further studies on eq.(32) and eq.(33), including how to decide σ^2 and experimental results, are given in (Xu, 1995b). In addition, the studies on the other cases given in sec.3 are provided in (Xu, 1995c).

4.3 Generalized Finite Mixture Models

When $r(\Psi_m) = -\frac{\beta}{\beta_r}\ln\frac{\alpha_m}{Z_m}$ with $\alpha_m = \frac{1}{N}\sum_{i=1}^{N}\omega_{mi}$, $Z_m = Z_m(\beta,\Psi_m) = \int exp(-\varepsilon^q(x_i,\Psi_m)/\beta)dx_i$, the problem of minimizing J^q_M is equivalent to the maximum likelihood learning of mixture distribution

$$p(x) = \sum_{m=1}^{N_c}\alpha_m p_m(x/\Psi_m), \quad p_m(x/\Psi_m) = \frac{1}{Z_m}exp(-\varepsilon^q(x_i,\Psi_m)/\beta).$$

This sets up connections of MML learning to finite mixture and EM learning (Dempster, Laird & Rubin, 1977, Hathaway, 1986). The connection between EM and statistical physics is also explored recently by Yuille, Stolorz and Utans (1994). Via MML learning and particularly eq.(33), we can further connect the EM learning to MDL learning. In addition, the 2nd term in eq.(9) was used in Yuille and Kosowsky (1992) as barrier function; It suggests that MML learning may also be connected to optimization.

5. Conclusions

A unified learning framework is proposed. It has been shown that the different special cases of this framework will automatically led us to current existing major types of neural network learnings, e.g, data clustering, various PCA-type self-organizations and their localized extensions, self-organizing topological map, as well as supervised learning for feedforward network and modular architecture. With this new framework, we can get a deep understanding on the existing learnings, we can also be guided to extend the existing learnings for new topics and results. We have laid a path for introducing regularization or generalization to various unsupervised learnings, particularly, we have developed approaches for a classical unsolved problem—how to decide the number of clusters in clustering analysis or competitive learning.

References

A.P.Dempster, N.M. Laird & D.B.Rubin (1977), *J. of Royal Statistical Society, B39*, 1-38.

R.J.Hathaway(1986), *Statistics & Probability Letters 4*, 53-56.

M.I.Jordan & R.A.Jacobs (1994), *Neural Computation 6*, 181-214.

M.I.Jordan & L. Xu (1995), Convergence results for the EM approach to mixtures-of-experts architectures, to appear on *Neural Networks.*

T. Kohonen (1989), *Self-Organization and Associative Memory*, 3rd ed. Springer-Verlag.

T.K.Leen & N.Kambhatla(1994), in *Advances in NIPS 6*, 152-159.

E. Oja(1989), *Int. J. of Neural Systems*, vol. 1, 61-68.

L. Xu(1991), In *Proc. IJCNN'91*, Singapore, Nov. 1991, 2362-2373.

L. Xu & E. Oja(1992), *Neural Networks*, vol. 5, 441-457.

L. Xu(1993), *Neural Networks*, vol. 6, 627-648.

L.Xu & M.I.Jordan(1993), *Proc. WCNN'93, Portland*, OR, Vol. II, 431-434.

L.Xu (1994a), *Proc. IEEE ICNN'94, Orlando*, (invited paper), Vol.I, 315-320.

L.Xu (1994b), *Proc. IEEE ICNN'94, Orlando*, (invited paper), Vol.II, 1252-1257.

L.Xu (1994c), *Proc. ICONIP94, Seuol*, (invited paper), 943-949.

L. Xu, M.I. Jordan & G.E. Hinton (1995), An alternative model for mixtures of experts, to appear on *Advances in Neural Information Processing Systems 7*, eds., Cowan, J.D., Tesauro, G., and Alspector, J., MIT Press, 1995.

L.Xu & M.I.Jordan(1995), On convergence properties of the em algorithm for gaussian mixtures, to appear on *Neural Computation.*

L.Xu (1995a), Local and hierarchical LMSER learning for vector quantization, to appear.

L.Xu (1995b), Deciding number of clusters in clustering analysis and competitive learning, to appear.

L.Xu (1995c), Model selection for LMSER, PCA and their localized versions, to appear.

A.L. Yuille, P. Stolorz & J.Utans (1994), *Neural Computation 6*, 334-340.

A.L. Yuille & J.J. Kosowsky, Harvard Robotics Lab. Tech. Rep. 92-7. 1992.