An Overview on Unsupervised Learning from Data Mining Perspective

Lei Xu *

Department of Computer Science and Engineering, Chinese University of Hong Kong, Shatin, NT, Hong Kong, P.R. China, E-mail: lxu@cse.cuhk.edu.hk

Advances in Self-Organising Maps: WSOM 2001 Proceedings
Nigel Allinson, et al eds, Springer, June 2001, pp181-210

Abstract. A number of unsupervised learning studies have been summarized from the perspectives of (a) mining dependence among components (PCA and MCA, LMSER and nonlinear PCA, ICA and independent factor analyses, three layer net and hidden unit number, etc); (b) mining homogeneous or nonhomogeneous groups among samples (MSE clustering and VQ, RPCL learning and elliptic extensions, Gaussian mixture and modified EM algorithms, etc); (c) mining dependences within local groups (localized extensions of part (a) as well as modular supervised learning such as RBF nets, mixture-of-experts and kernel on support vectors); and (d) mining topological structures with organized groups (self-organization map and others).

1 Data Mining and Statistical Learning

In recent years, knowledge database discovery (KDD) becomes an increasingly popular area. Generally, it stands for discovering and visualizing the regularities, structures and rules from data. Its core part [22, 28, 8] consists of dimension reduction, structure mining and visualization, which is, referred as *data mining* (DM) in a broad sense, actually a revisit from a more engineering perspective of those statistical learning studies in the literature of neural networks and pattern recognition [108, 20, 19, 21].

The so called regularities, structures and rules are actually different expressions of various types of dependences among data. We can categorize these types of dependences in five levels based on the complexities of data in different media. Learning from data is actually made through samples of data. Usually, a sample is in the expression of a vector $x = [x^{(1)}, \cdots, x^{(d)}]^T$ with each component $x^{(i)}$ called feature that is either a real number or a symbol. Thus, the basic level of dependences is among features and can be represented in a set of either mathematical functions or logical rules. The next level of dependences is demonstrated via the locations of samples vectors. That is, under such dependences, samples located closely are regarded as same or similar and thus form a group

* The work described in this paper was fully supported by a grant from the Research Grant Council of the Hong Kong SAR (project No: CUHK4383/99E).

or cluster, such that each sample in a cluster can be represented by either any one sample in this cluster or a representative of this cluster (e.g., by the cluster center) [35]. In contrast, samples that locates far away are different and belong to different clusters. In other words, this level of dependence participates samples into a number of equivalence class that is called groups or clusters. Again, such a type of dependences can be represented via either mathematical functions or logical rules. The third level of dependence is indicated via the topological links among clusters, which is usually demonstrated via the topological links of centers of clusters [73, 42]. Moreover, the next level of dependences is temporal dependences among data that come from speech, audio or other time series [30]. Finally, for data from image and video, we have the 5th level of dependences that represents spatial relations of samples. The spatial relations can be in either the form of regular 2D or 3D lattice or of the more general graphic topology such as a probabilistic graphical structure or called Bayesian or Belief networks [68, 36].

A specific learning task may focus on learning or mining one or a combination of the five levels of dependences, and thus we can have a number of learning tasks in different complexities. In this paper, we focus on the first three levels of dependences, which lead to five meaningful combinations that are referred as learning tasks. Shortly we denote them as Task 1, Task 2, Task 1-2, Task 1-3, Task 1-2-3, where Task 1 means mining the first level of dependences and Task 1-2 means mining a combination that consists of the first level and second level of dependences. Specifically, we explore these tasks from statistical perspective. That is, data is regarded to consist of independent and identically distributed (i.i.d.) samples from a underlying unknown distribution $p(x)$ with each sample being a vector $x = [x^{(1)}, \cdots, x^{(d)}]^T$, and each of the tasks actually involves estimating either the distribution $p(x)$ or its certain statistics.

In the sequel, we provide an overview on several typical statistical learning models and algorithms that implement the five learning tasks. There are already certain survey papers in the literature. Instead of providing a complete overview on all the existing studies in the literature, we focus those studies which have not been well discussed yet, and particularly on those studies that can be viewed from a unified perspective. Moreover, we also emphasize the issues of regularization and model selection for the well known problem of a small size of samples.

2 Task 1: Mining Dependence among Components

2.1 Typical Dependence Structures

We consider three categories of typical dependence structures as follows.

(1) Forward Mapping Dependence among components are represented via mapping x forwardly to another vector $y = [y^{(1)}, \cdots, y^{(m)}]^T$. Specifically, this forward mapping $f(x, \theta) = y$ can be further classified in three types according to the role of y in representation, namely, whether the role of y is minimized or maximized:

• *Pattern Matching* The idealistic case that the role of y is minimized happens at $f(x,\theta) = y = 0$ for all the samples of x. It means that these samples match the pattern described by $f(x,\theta) = 0$ with a specific value of θ, where $f(x,\theta) = 0$ is a set of either mathematical functions or logical rules. Usually, $y \neq 0$ and is regarded as random error or residual under this matching. It is natural to assume that the residual y should be unbiased $Ey = 0$, and that no dependence relation remains among components, e.g., the components become uncorrelated $Eyy^T = \Lambda = diag[\lambda_1, \cdots, \lambda_m]$ in a sense of independence up to the 2nd order statistics, where Eu denotes the mean of u and $diag[\lambda_1, \cdots, \lambda_m]$ denotes a diagonal matrix that consists of diagonal elements $\lambda_1, \cdots, \lambda_m$. In other words, we can model the residual y by a Gaussian $q(y|\theta_y) = G(y|0, \Lambda)$, where and in the sequel, $G(u|m, \Sigma)$ denotes a Gaussian density with mean m and covariance matrix Σ. Generally, we consider that the components of residual y are completely independent:

$$q(y|\theta_y) = \prod_{j=1}^{m} q(y^{(j)}|\theta_y^{(j)}). \tag{1}$$

To minimize the role of y, we also expect that the variance of each $y^{(j)}$ is minimized. To make this desire meaningful, we usually need certain constraint on θ, shortly denoted by $S(\theta)$. E.g., when $f(x,\theta)$ is linear as given by

$$f(x,\theta) = \theta g(x), g(x) = \begin{cases} [x,1]^T, & \text{(a)}, \\ \text{a nonlinear map of } x, & \text{(b)}, \end{cases} \tag{2}$$

if we have $Ey = 0$ and $Eyy^T = \Lambda$ diagonal for $\theta g(x) = y$, then for any diagonal D and orthogonal ϕ with $\phi^T\phi = I$, we still have that

$$Ey' = 0,\ Ey'y'^T = D^2\ for\ y' = D\phi\Lambda^{-0.5}y. \tag{3}$$

We can remove this indeterminacy by imposing the constraint $\theta^T\theta = I$ as $S(\theta)$. Then, we can learn θ via

$$\min_{\theta,\ s.t.\ \theta^T\theta=I} \frac{1}{N}\sum_{t=1}^{N}\|y_t\|^2,\ y_t = \theta g(x_t), \tag{4}$$

or equivalently we can regard the residual y_t being Gaussian and implement the following ML learning

$$\max_{\theta,\Lambda,\ s.t.\ \theta^T\theta=I} \frac{1}{N}\sum_{t=1}^{N} \ln G(y_t|0,\Lambda),\ y_t = \theta g(x_t). \tag{5}$$

Generally, for eq.(1) we further have

$$\max_{\{\theta,\theta_y,\ s.t.\ S(\theta)\}} \frac{1}{N}\sum_{t=1}^{N} \ln q(y_t|\theta_y),\ y_t = f(x_t,\theta). \tag{6}$$

• *Maximum Information Transfer* In contrast, to maximize the role of y means to make $f(x,\theta)$ able to transfer maximum information from x. Typically, we expect to maximize the variance of every components of y. However, it may also

not meaningful, if there is no some extra constraint $S(\theta)$. E.g., when $f(x,\theta) = \theta g(x_t)$, each variance tends to infinity for any $c\theta$ as $|c| \to \infty$. This problem can be removed again by the constraint $\theta^T\theta = I$. Without losing any generality, we can assume $Ey = 0$ (otherwise we can simply let $f(x,\theta) - Ey$ to be a new forward mapping). In this case, we have

$$\max_{\theta,\ s.t.\ \theta^T\theta=I} Tr[\Sigma_y] \quad or \quad \max_{\theta,\ s.t.\ \theta^T\theta=I} |\Sigma_y|,$$
$$\Sigma_y = \frac{1}{N}\sum_{t=1}^{N} y_t y_t^T, \quad y_t = \theta g(x_t). \tag{7}$$

Generally, we can maximize the information transferred from x via $f(x,\theta)$ by

$$\max_{\{\theta,\theta_y,\ s.t.\ S(\theta)\}} [-\int p(y)\ln p(y)dy],\ p(y) = \int p(y|x)p_0(x)dx, \tag{8}$$
$$y = f(x,\theta)\ or\ p(y|x) = \delta(y - f(x,\theta)),\ p_0(x) = \frac{1}{N}\sum_{t=1}^{N}\delta(x - x_t),$$
$$E.g.,\quad S(\theta)\ is\ \theta^T\theta = I,\ for\ y = \theta g(x).$$

Particularly, for the linear mapping

$$y = \theta x + m_y, \quad \theta^T\theta = I, \tag{9}$$

when x is Gaussian, then y is also Gaussian. In this case, we have that eq.(8) degenerates back to eq.(7). Alternatively, changing 'max' into 'min' in eq.(5) will also become equivalent to eq.(7).

In contrast to the case that the role of y is minimized as residuals such that the dependence structure among components are directly represented in $f(x,\theta) = 0$, it looks not so directly how $f(x,\theta) = y \neq 0$ explores the dependence structures when the role of y is maximized. This issue can be understood in two different situations. On one hand, when the mapping by $f(x,\theta) = y$ is information preserving with all the information of x transferred to y, there is no need to drive $f(x,\theta)$ to explore the structure of samples of x for this information transfer. A typical example is eq.(9) with $|\theta| = I$. In this case, we simply have $Tr[E(\theta x x^T\theta^T)] = Tr[E(xx^T)]$, $|E(\theta x x^T\theta^T)| = |E(xx^T)|$, and even $-\int p(y)\ln p(y)dy = -\int p(x)\ln p(x)dx$. Thus, all the information is preserved for any θ with $|\theta| = I$, irrelevant to the structure of samples of x. $|\theta| = I$. On the other hand, when $f(x,\theta) = y$ can only implement an information loss mapping, in order to maximize the information transferred $f(x,\theta) = y$ must explore the major structure of x such that the major information part is picked to map. In this case, eq.(7) or eq.(8) will make y become not only independent in components, and but also carry the major information of x.

• *Equalization* In the above two mapping types, $f(x,\theta) = y$ acts in two aspects, namely making y become decorrelated or independent in components, and making the role of y either minimized or maximized. Alternatively, we also have a third choice that we only consider the first aspect. In this case, we replace the constraint $S(\theta)$ such as $\theta^T\theta = I$ in eq.(9) by imposing the constraint that every

component of y is distributed under a same density, e.g., we impose $Eyy^T = I$ in eq.(9). We call this 3rd type of mapping *equalization*, as it can be regarded as an extension of histogram equalization used in image processing. Clearly, to reach such an equalization, $f(x, \theta) = y$ must be able to explore the major structure of x unless x has no structure among components.

(2) Backward Mapping In contrast to a forward mapping, we can also consider a backward mapping $g(y, \phi) = x$ which represents how y is mapped backward to generate x. Usually, a Gaussian noise e is taken into consideration, and thus it becomes $g(y, \phi) + e = x$. More generally, this mapping is represented by

$$q(x|\theta) = \int q(x|y, \theta_{x|y}) q(y|\theta_y) dy. \tag{10}$$

E.g., $g(y, \phi) + e = x$ under Gaussian e gives $q(x|y, \theta_{x|y}) = G(x|g(y, \phi), \Sigma_e)$ where the covariance matrix Σ_e is either simply $\sigma_e^2 I$ or diagonal. Moreover, $q(x|y)$ can be an other density when e is nonGaussian. Also $q(y|\theta_y)$ is involved in eq.(10).

The key point for learning the parameters of $\theta_{x|y}, \theta_y$ is to make $q(x|\theta)$ fit a given set of samples of x under a learning principle. The typical example is the ML learning

$$\max_{\{\theta_{x|y}, \theta_y, \ s.t. \ S(\theta)\}} \frac{1}{N} \sum_{t=1}^{N} \ln p(x_t|\theta), \tag{11}$$

where $S(\theta)$ is again certain constrain to be impose on θ to avoid certain indeterminacy that depends what type of $q(y|\theta_y)$ is used. In this paper, we consider the following three types of $q(y|\theta_y)$ that trades off the strengths of $q(x|y, \theta_{x|y}), q(y|\theta_y)$ in modeling the dependence structure of samples of x:

• *Homogenous Independent Factors* The simplest type is each component, or called factor, of y being a standard Gaussian, i.e., $G(y|0, I)$, while $q(x|y, \theta_{x|y})$ takes all the responsibility to describe the dependence structure. As to be discussed in Sec.2.4, the disadvantage of the simplest type is leading to certain crucial indeterminacy. Another type is that each component density $q(y^{(j)}|\theta_y)$ in eq.(1) is a same non-Gaussian density, which provides a complementary help to $q(x|y, \theta_{x|y})$ in modeling the dependence structure of data.

• *Nonhomogenous Independent Factors* More generally, $q(y^{(j)}|\theta_y)$ may be different for different j such that $q(y|\theta_y)$ takes more share on modeling the dependence structure of data. E.g., we consider $q(y^{(j)}|\theta_y)$ that is same to every j only in its first two order statistics but different in higher order statistics, as given by an expansion

$$q(y^{(j)}|\theta_y^{(j)}) = G(y^{(j)}|0, 1)[1 + k_3(y^{(j)})h_3(y^{(j)})/6 + k_4(y^{(j)})h_4(y^{(j)})/24], \tag{12}$$

where k_3, k_4 are the cumulants of $y^{(j)}$, and h_3, h_4 are Hermite polynormails. Another example is that each $q(y^{(j)}|\theta_y)$ in eq.(1) is given by a finite mixture as in [88].

• *Gaussian Mixture Factor* Instead of the independent factor model eq.(1), another interesting type of $q(y|\theta_y)$ is either a mixture of gaussians $G(y, \mu_j, I)$ with different location μ_j or a nonhomogenous Gaussian mixture with each Gaussian with a different covariance matrix.

(3) Forward Mapping and Backward Mapping We can combine the features of both the forward mapping and backward mapping. There are two ways for such a combination. One is to make learning on each separately and then put them together. The other is to consider the both mappings coordinately during parameter learning. The details will be further discussed later.

Finally, we add that either $f(x, \theta)$ or $g(y, \phi)$ may also be trained via supervised learning algorithms when a set of paired samples $\{x_t, y_t\}$ is known. Supervised learning tasks are encountered in pattern recognition, prediction, function approximation, discriminant analysis and many others. A number of survey papers on these issues are available in the literature. E.g., an early systematical overview is referred to [108]. This paper will focus on unsupervised learning. However, by considering $\xi_t = \{x_t, y_t\}$ as an augmented sample vector, a supervised learning task can also be regarded as a unsupervised learning task on ξ_t under the structural constraint between its part x_t and its part y_t. For this reason, this overview also includes in appropriate cases certain results on supervised learning. Meanwhile, we only need to consider how to get $f(x, \theta)$ based on a training set of $\{x_t, y_t\}$ since the task of getting $g(y, \phi)$ is similar.

2.2 Forward Mapping (I): Maximum Information Transfer

• *PCA and Subspace Analyses* Studies on this topic can be traced back as early as [32]. In past decades, Principal Component Analysis (PCA), obtained directly from eq.(7) with $y = Wx$, has been widely studied and used in the field of statistics and many other fields. Its close relation to the well known Hebbian learning in neuroscience was first established in [62] where a simple linear neuron trained by an adaptive modified Hebbian learning rule is shown to perform exactly PCA. Since then, extensive studies have been made to develop adaptive algorithms on linear neural nets of multiple units for PCA. The studies consist of two main streams. One is for adaptive algorithms on asymmetrical architecture that extract the first k-PCA vectors sequentially one by one, with an initial study made in [74] and several sequential studies referred to [45]. The other stream consists of adaptive algorithms on symmetrical architectures that extract the first k-PCA vectors in parallel. Oja subspace rule [63] is one initial effort, which as well as several related efforts, however, do not perform k-PCA as discussed in [103], but a "collective version" of PCA, called *Principal Subspace Analysis (PSA)* that extracts a subspace spanned by the k-PCA vectors.

Two years after [63], two further results have been obtained by the present author in [103]. One is that a global convergence proof on the Oja subspace rule was provided. The other is an adaptive algorithm is given for extracting the m PCA vectors instead of only performing PSA. Moreover, adaptive PCA

algorithms for robust performance and for working on missing data have also been proposed [94, 99]. Furthermore, in the past decade, extensive studies have also been made on showing that PCA or PSA can be equivalently implemented by various different learning rules and architectures. A number of such criteria and the corresponding adaptive learning rules are summarized in 1994 [100, 101]. For examples, PCA is performed by either of the following two criteria

$$(a)\ \max_W E||y^T \Lambda^{-1} y||^2,\ (b)\ \max_W E||x - W^t y||^2,\ y = Wx,\ s.t.\ WW^t = I, \quad (13)$$

where $\Lambda = diag[\lambda_1, \cdots, \lambda_m],\ \lambda_1 > \cdots > \lambda_m$ is pre-given. The case (a) is called *Maximum Variation of Normalized Output (MVNO)* and the case (b) is called *Least Mean Square Error Reconstruction (LMSER)*. We can get adaptive learning to implement them as shown in [101].

Typical PCA applications relate to compressing and representing as well as visualizing data in a reduced dimension. Also, PCA is used in [106] for retrieving complicated structures via an approximate but fast implementation of attributed graph matching. Moreover, as a dual to [107], the direction of the first principal vector provides a total least square fitting of data by a line. Also, we can approximately fit a curve by transferring it to a line, e.g., $g(x)$ in eq.(2) is set by the same trick in [107].

• *Nonlinear PCA and ICA* PCA has two main features, namely the mapping is linear $y = Wx$ and the components of y are uncorrelated (i.e., independent only in the 2nd order statistics). Extensions have been made on changing each of the features. Under the name of *nonlinear PCA*, the efforts on extending $y = Wx$ into the post-linear $y = S(Wx)$ with $S(y) = [s(y(1)), \cdots, s(y^{(m)})]$, in a sense that each component of linear mapping Wx is followed by a nonlinear scaling by a scalar function (e.g., a sigmoid function). Two typical efforts were made in 1991 from different perspectives. One is to use $y = s(Wx)$ in Oja rule and it turned out that the learning becomes very robust under noise and outliers [64]. The other is made in [103] (published firstly in Proc. IJCNN91), that uses $y = s(Wx)$ in the above $E||x - W^t y||^2$ in eq.(13) with an adaptive gradient algorithm provided in its Eq.(9b), which is rewritten as follows:

$$\min_W E||x - W^t S(Wx)||^2,\ W^{new} = W^{old} + \eta[S(Wx)e^T + S'(Wx)\varepsilon x^T], \quad (14)$$
$$e = x - W^t S(Wx),\ \varepsilon = y - WW^t S(Wx), S'(y) = diag[s'(y(1)), \cdots, s'(y^{(m)})],$$

where $s'(r) = ds(r)/dr$. Moreover, it has been firstly discovered experimentally that the sigmoid nonlinearity $s(r)$ can automatically break the symmetry in the PSA subspace such that components of W tend to directions that are close to but not equal to the first m orthogonal PCA directions. Also, the other PCA criterion $E||y^T \Lambda^{-1} y||^2$ in eq.(13) is also extended into the so called the Nonlinear Maximum Variance (NMV) learning rule [96] by using the post-linear unit $y = s(Wx)$.

Started from Jutten and Herault in 1988 [39], under the name *Independent Component Analysis (ICA)*, efforts have been made on targeting on making

the components of $y = Wx$ independent beyond the 2nd order statistics. Due to this clear motivation, it has become a popular topic in the past decade. Advances on this topic can be roughly classified into four streams. One is based on solving the equation group on the higher order moments of x and of y subject to $y = Wx$ and the independent components of y. The second is to minimize a so called contrast function such that the minimum is reached when y becomes independent, with the minimization implemented by adaptive algorithms. The third one is the above mentioned nonlinear PCA. Actually, three years after its publication [103], eq.(15) was directly adopted to implement ICA by authors of [40] with some promising results, which experimentally sets up a link between ICA and nonlinear PCA by eq.(15). The last one is called information theoretic approach, which is further discussed in the sequel.

Typical ICA applications include blind source separation, feature extraction, medical information processing, and many others. Extensive publications on ICA are also available in literatures, e.g., readers are referred to a special issue [27] and a survey paper [33]. Here we only further provide certain advances on the information theoretic approach, roughly summarized into three stages. The first stage is featured by the INFORMAX [5] and the MMI [2]. The two typical algorithms as well as several others update W adaptively after the marginal density of each components of y is heuristically prefixed [5] or estimated via density expansion [2]. These algorithms work well on the cases that the components of y are either all sub-Gaussians or all super-Gaussians, but fail on the cases that the components of y consist of partly sub-Gaussians and partly super-Gaussians. At the second stage, it is realized [91] that the marginal density $q(y^{(j)})$ of each component should also be learned simultaneously during learning on W to automatically fit any combination of super-Gaussian or sub-Gaussian components of y. This idea could be implemented by learning the parameters θ_y in eq.(1). An early effort is the learning parametric mixture based ICA [91, 88], where a finite mixture is used as $q(y^{(j)}|\theta_y^{(j)})$ and $\theta_y^{(j)}$ is updated by an EM-like algorithm during updating W. Alternatively, efforts in [65, 15] estimate the kurtosis of $q(y^{(j)})$. At the third stage, extensions have been made towards various cases, e.g., (a) the dimension of x is larger than that of y instead of that W is invertible [89, 88, 86], (b) some specific nonlinear ICA [78, 86], (c) the so called temporal ICA that takes temporal relation among samples in consideration [82] and (d) the so called competitive ICA, which will be further discussed in Sec.4.1.

• *Supervised learning: discriminant analysis and three layer net* Forward mapping can also be built via supervised learning on a training set of paired samples $\{x_t, y_t\}$. A typical example of linear mapping $y = Wx$ is given by the Fisher discriminant analysis that maps x into y for a best classification purpose [23, 21].

The key advantage of a forward mapping via supervised learning is able to explore the complicated nonlinear mapping among given samples in pairs $\{x_t, y_t\}$, in help of various forward networks. Readers are referred to [108] for an overview. Here, we only introduce certain new results obtained in recent years

from Bayesian Yang-Ying learning [80, 82, 83] on the following three layer net for mapping $x \to \xi \to y$:

(a) $x \to \xi$ by a post-linear map $s(Ax+a_0)$ and ξ from a multivariate Bernoulli $q(\xi|x) = \prod_{j=1}^m s(z^{(j)})^{\xi^{(j)}}[1-s(z^{(j)})]^{1-\hat{\xi}^{(j)}}, z = Ax + a_0, s(r) = 1/(1+e^{-r})$;

(b) $\xi \to y$ is made by linear mapping $B\xi + b_0$ subject to a Gaussian noise with covariance $\sigma^2 I$, i.e., $q(y|\xi) = G(y|B\xi + b_0, \sigma^2 I)$.

Thus, the mapping $x \to y$ is given by $q(y|x) = \sum_\xi G(y|B\xi + b_0, \sigma^2)q(\xi|x)$. By the linear Taylor expansion of $G(y|B\xi + b_0, \sigma^2)$ around the mean $E\xi = s(Ax+a_0)$, we approximately have $q(y|x) \approx G(y|Bs(Ax+a_0)+b_0, \sigma^2)$, and thus $E(y|x) = \int yq(y|x)dy = Bs(Ax + a_0) + b_0$ implements the conventional three layer net with sigmoid hidden units.

Instead of using the back-propagation technique to train this three layer net in the sense of least square learning, we make its training via the so called BYY harmony learning [81], which provides the following EM-like algorithm:

$$
\begin{aligned}
&E\ step:\ \hat{\xi} = arg\max_{\xi}\{G(y|B\xi + b_0, \sigma^2 I)\prod_{j=1}^m s(z^{(j)})^{\xi^{(j)}}[1-s(z^{(j)})]^{1-\xi^{(j)}}\},\\
&M\ step:\ (a)\ e_y = y - B^{old}\hat{\xi} - b_0^{old},\ B^{new} = B^{old} + \eta e_y \hat{\xi}^T,\\
&\qquad b_0^{new} = b_0^{old} + \eta e_y,\ \sigma^{2\ new} = (1-\eta)\sigma^{2\ old} + \eta\|e_y\|^2, \qquad (15)\\
&\qquad (b)\ e_\xi = \hat{\xi} - s(Ax + a_0),\ A^{new} = A^{old} + \eta e_\xi x^T,\ a_0^{new} = a_0^{old} + \eta e_\xi.
\end{aligned}
$$

Specifically, E-step is equivalent to find $\hat{\xi}$ that maximizes $C(\xi) = -0.5\sigma^{-2}\|y - B\xi - b_0\|^2 + \sum_{j=1}^m [\xi^{(j)} \ln s(z^{(j)}) + (1-\xi^{(j)}) \ln (1 - s(z^{(j)}))$, which is a typical discrete quadratic programming problem. For a fast approximation, we can first solve the linear equation $\nabla_\xi C(\xi) = 0$ by regarding ξ being real and then hard-cutting the solution into a binary one. That is,

$$
\begin{aligned}
&\hat{\xi}^{(j)} = \begin{cases} 1, & \text{if } \bar{\xi}^{(j)} > 0.5, \\ 0, & \text{otherwise,} \end{cases} \quad \bar{\xi} = (B^T B)^{-1}[B^T(y - b_0) + \sigma^2 \pi],\\
&\pi = [\pi^{(1)}, \cdots, \hat{\pi}^{(m)}]^T,\ \pi^{(j)} = \ln \frac{s(z^{(j)})}{1 - s(z^{(j)})}. \qquad (16)
\end{aligned}
$$

Another issue is to set up the step size η in the M-step. It can be two choices:

(a) It is fixed at a small constant $\eta > 0$ when the learning eq.(15) is made in the so called *empirical learning* [81]. Moreover, its specific value can be different for different parameters. E.g., two different sizes η_A, η_B are used for updating A, B, respectively.

(b) When the learning eq.(15) is made in the so called *normalization learning* [81], it is given by $\eta = \eta_t \eta_0$ with $\eta_0 > 0$ and η_t given by

$$
\begin{aligned}
&\eta_t = \frac{1}{N} - \gamma_0 \frac{\gamma_t}{S_q},\ \gamma_t = G(y|B\hat{\xi}_t + b_0, \sigma^2 I)\prod_{j=1}^m s(z^{(j)}(t))^{\hat{\xi}_t^{(j)}}[1 - s(z^{(j)}(t))]^{1-\hat{\xi}_t^{(j)}},\\
&S_q = \sum_{t=1}^N G(y|B\hat{\xi}_t + b_0, \sigma^2 I)s(z^{(j)}(t))^{\hat{\xi}_t^{(j)}}[1 - s(z^{(j)}(t))]^{1-\hat{\xi}_t^{(j)}}, \qquad (17)
\end{aligned}
$$

where $1 > \gamma_0 > 0$ is a given constant that compensates the finite sample size in normalization by $\sum_{t=1}^N q(y_t|\hat{\xi}_t)q(\hat{\xi}_t|x_t)$. By this η_t, after the winner-take-all

competition by the E-step, a de-learning is introduced to regularize the learning on the winner for each sample in proportional to the current fitting of the model to the sample. However, it is expensive to compute on all the samples as in eq.(17). We can also approximate the sum S_q adaptively by $S_q(t+1) = (1-\lambda)S_q(t) + \lambda G(y|B\hat{\xi}_t + b_0, \sigma^2 I)\prod_{j=1}^{m} s(z_t^{(j)})^{\hat{\xi}_t^{(j)}}[1-s(z_t^{(j)})]^{1-\hat{\xi}_t^{(j)}}$ for a suitable $0 < \lambda < 1$. Then, as t varies, we have

$$\eta_t = \frac{1}{t} - \gamma_0 \frac{\gamma_t}{S_q(t+1)}. \tag{18}$$

The detail derivations of eq.(15), eq.(16) and eq.(17) are further referred to [81], from which we also known that eq.(15) is actually implement a so called harmony learning that will push $p(\xi|x)$ into a least complexity form to avoid using extra hidden units. Alternatively, by enumerating a number of m values incrementally, we can also select a best number m^* for hidden units by the following criterion

$$\begin{aligned} &m^* = arg\min J(m),\ J(m) = 0.5m\ln\sigma^2 + J_y(m),\ \hat{\xi} = Ax + a_0, \\ &J_y(m) = -\tfrac{1}{N}\textstyle\sum_{t=1}^{N}\sum_{j=1}^{m}\{\hat{\xi}_t^{(j)}\ln s(z_t^{(j)}) + (1-\hat{\xi}_t^{(j)})\ln(1-s(z_t^{(j)}))\}. \end{aligned} \tag{19}$$

2.3 Forward Mapping (II): Pattern Matching and Equalization

• *Pattern Matching: MCA-MSA and Surface Fitting* Playing a dual role to PCA, eq.(4) results in the so called minor component analysis (MCA) that minimizes the residuals y such that the samples are fitted in the total least square sense by either the hyperplane with the 1st minor component as its normal direction or the subspace that is orthogonal to the subspace spanned by the minor components. That is, using a m-dimensional MSA subspace to represent a data set is equivalent to use a $d-m$ dimensional PSA subspace to represent to represent the same set, where n is the dimension of x. Particularly, instead of using $d-1$ principal components to represent a hyperplane, we can equivalently represent it by only the 1st minor component. In [109], MCA or MSA is used together with PCA or PSA to form a dual representation with a more effective dimension reduction. In 1992, an adaptive learning algorithm has been proposed to implement MCA for the total least square fitting of not only lines, planes, and hyperplanes but also circles, curves, planes, surfaces, and hypersurfaces [107]. Moreover, MCA is also used in [87] for the so called co-integration regularity in time series. Furthermore, studies has also been extended to object identification via fitting a general function $g(y, \phi)$ [95].

• *Equalization: ICA, P-ICA and M-ICA* Though the concept of ICA was proposed in parallel to PCA, the existing studies on ICA are made without distinguishing the concepts 'minor' or 'principal'. There may be two reasons. One is that the original purpose of ICA is to recover $x = Ay$ by $y = Wx$ and such a recovery is indeterminacy on the scales of each component of y. The other is that W is usually invertible and $y = Wx$ becomes independent but take all

the components in consideration, where to distinguish which components are 'minor' or 'principal' is not necessary.

Strictly speaking, the concept of ICA is parallel to de-correlation component analysis (DCA) $y = Wx$ that makes Eyy^T be diagonal, including both PCA and MCA as well as linear equalization by $y = Wx$ with $Eyy^T = I$ as special cases. Thus, when the dimension m of y is lower than the dimension d of x, we should also have the principal ICA (P-ICA), minor ICA (M-ICA), and nonlinear equalization. The last one has already discussed in Sec.2.1. The difference of the principal ICA and minor ICA can be understand from the perspective of the so called pre-whitening ICA. After making the covariance of samples of x be orthogonal, we will have C_d^m combinations to select m de-correlated components and then normalize them white. Thus, ICA can be made in two steps. First, we select a combination of m components and then normalize them white. Second, we do the invertible ICA in the m dimension space. That is, we have C_d^m combinations of ICA. Among them, we have either the P-ICA when the m principal components are used to whitening or the M-ICA when the m minor components are used to whitening. This way of defining the P-ICA and the M-ICA depends on the whitening preprocessing. In Sec.2.5, we will give another way to define them without relying on a preprocessing.

2.4 Backward Mapping: Three Typical Independent FA

• *Gaussian FA and Independent FA* The typical example of the early efforts on this topic is factor analysis, which can be traced back to the beginning of the 20th century by Spearman [77]. Formulated by Anderson and Rubin in 1956 [3], it considers the simplest linear special case of eq.(8), that is

$$x = Ay + e, \quad e \text{ is independent from } y, \tag{20}$$

where both e, y come from Gaussians with $E(e) = 0, E(y) = 0, E(y_t y_t^T) = I$, and e is uncorrelated among its components with a diagonal covariance matrix Σ_e. However, this model suffers the problem of not having a unique solution because its indeterminacy on rotation and on the communality estimation [55]. Early studies towards such problems consists of either constraining A to be orthogonal matrix only or imposing heuristics to select a specific rotation such as in Quartimax and Varimax [55].

In the past decade, efforts have been made on considering eq.(20) with the independence assumption on the components of y. For clarity, we refer this new type of factor analysis (FA) as *independent FA (IFA)* to avoid being confused with the original one, which should now be more precisely referred as *De-correlating FA (DFA)*. Similar to ICA, when each $q(y^{(j)})$ is nonGaussian or at most only one of them is Gaussian, the rotation indeterminacy can be removed. However, it is much more difficult to implement IFA than ICA. Not only we need to deal with the problem of modeling each component density $q(y^{(j)})$ as in ICA, but also we need to handle the noise e. Due to this noise, the ML learning by eq.(11) encounters the computational difficulty on handling the integral

over y in eq.(8). Several efforts have been made towards to solving this difficulty. The most simple way is to approximately regard $e = 0$ such that estimating A in $x = Ay$ becomes equivalent to ICA that gets the inverse mapping $y = Wx$. But it works only in a small noise e case. The other ways include preprocessing for filtering noise, making ML learning via a Monte-Carlo sampling [86], and using heuristic structures [15,27]. Readers are referred to a survey paper [33].

In the sequel, we further add on certain advances obtained from BYY harmony learning in recent years [80, 82, 83].

• *Bernoulli FA, Independent FA, and BYY harmony learning* We consider eq.(20) with eq.(1). In help of the BYY harmony learning [80], we can get the following EM-like algorithm:

$$\begin{aligned} &E\ step:\ \hat{y} = arg\max_y[G(x|Ay, \Sigma_e)q(y|\theta_y)], \\ &M\ step:\ e = x - A^{old}\hat{y}, A^{new} = A^{old} + \eta e\hat{y}^T,\ \Sigma_e^{new} = (1-\eta)\Sigma_e^{old} + \eta ee^T, \\ &\qquad \theta_y^{new} = \theta_y^{old} + \eta\phi(\hat{y}),\ \phi(y) = \nabla_y \ln q(y|\theta_y). \end{aligned} \tag{21}$$

Specifically, E-step is equivalent to find $\hat{y}$ that maximizes $C(y) = -0.5(x - Ay)^T\Sigma_e^{-1}(x - Ay) + \sum_{j=1}^k \ln q(y^{(j)}|\theta_y^{(j)})$, which can be made by a fast approximation that solves the equation $\nabla_y C(y) = 0$ as shown in [80]. For illustration, we provide two examples:

$$\begin{aligned} &(a)\ for\ \ q(y|\theta_y) = G(y|0, I),\ \ \hat{y} = [I + A^T\Sigma_e^{-1}A]^{-1}A^T\Sigma_e^{-1}x, \\ &(b)\ for\ \ q(y|\theta_y) = \prod_{j=1}^k q_j^{y^{(j)}}(1-q_j)^{1-y^{(j)}},\ \ \hat{y}^{(j)} = \begin{cases} 1, & \text{if } \hat{y}^{(j)} > 0.5, \\ 0, & \text{otherwise}, \end{cases} \\ &\hat{y} = (A^T\Sigma_e^{-1}A)^{-1}\{A^T\Sigma^{-1}x + [\pi_1, \cdots, \pi_k]^T\}, \pi_j = \ln\frac{q_j}{1-q_j}. \end{aligned} \tag{22}$$

With the case (a) in eq.(21), there is no need on updating θ_y, and eq.(21) actually is an adaptive algorithm for implementing DFA. For the case (b), the updating on θ_y is simply given by

$$q_j = 1/(1 + e^{c_j}),\ \ c_j^{new} = c_j^{old} + \eta(\hat{y}^{(j)} - q_j^{old}). \tag{23}$$

In this case, eq.(21) is an adaptive algorithm for implementing a Bernoulli FA. Also a variant of Bernoulli FA is given in [80] for the case that x is also binary from a multivariate Bernoulli. In the literature of neural networks, other efforts have been also made on modeling binary x (e.g., representing a binary image) by interpreting it as generated from binary hidden factor y with mutually independent bits. Typical examples include multiple cause models [75,16] and Helmholtz machine [17,31].

Similar to eq.(15), another issue in eq.(21) is the step size η in the M-step. Again, it can be either fixed at constants for implementing *empirical learning* or given by $\eta = \eta_t\eta_0$ with

$$\eta_t = \frac{1}{N} - \gamma_0\frac{\gamma_t}{S_q},\ \ \gamma_t = G(x_t|A\hat{y}_t, \Sigma_e)q_j^{\hat{y}_t^{(j)}}(1-q_j)^{1-\hat{y}_t^{(j)}},$$

$$S_q = \sum_{t=1}^{N} G(x_t|A\hat{y}_t\Sigma_e) q_j^{\hat{y}_t^{(j)}} (1-q_j)^{1-\hat{y}_t^{(j)}}, \quad (24)$$

for implementing *normalization learning* [80]. Still, we can use eq.(18) by adaptively updating $S_q(t+1) = (1-\lambda)S_q(t) + G(x_t|A\hat{y}_t, \Sigma_e) q_j^{\hat{y}_t^{(j)}} (1-q_j)^{1-\hat{y}_t^{(j)}}$. Again, similar to eq.(15), the harmony learning of eq.(22) will push $q(y|\theta_y)$ into a least complexity form to avoid using a redundant dimension [80].

• *Principal subspace dimension* From BYY harmony learning [80, 82, 83], we can also get the following criterion for selecting a best dimension m^*:

$$\begin{aligned}
&m^* = arg\min J(m), \;\; J(m) = 0.5 \ln|\Sigma_e| + J_y(m), \\
&J_y(m) = \begin{cases} m\ln(2\pi) + m, & (a) \\ -\frac{1}{N}\sum_{t=1}^{N}\sum_{j=1}^{m}[\hat{y}_t^{(j)} \ln q_j + (1-\hat{y}_t^{(j)}) \ln(1-q_j)], & (b) \\ -\frac{1}{N}\sum_{t=1}^{N}\sum_{j=1}^{m} \ln q(\hat{y}_t^{(j)}|\theta_y^{(j)}), & (c), \end{cases} \\
&(a)\; q(y|\theta_y) = G(y|0, I), \quad (b)\; q(y|\theta_y) = \prod_{j=1}^{m} q_j^{y^{(j)}} (1-q_j)^{1-y^{(j)}}, \\
&(c)\; q(y|\theta_y) \; in \; eq.(1), \qquad (25)
\end{aligned}$$

where the value $J(m)$ is obtained after parameter learning at each m as we enumerate a number of m values incrementally. Specifically, the case (b) determines the number of binary bits that is required for y, while the case (a) and case (c) describe the dimension of the subspace dimension spanned by principal components in the sense of the 2nd order independence and of the higher order independence, respectively. E.g., we can use the case (a) together with PCA for determining an appropriate number m of principal components, and we can use the case (c) together with ICA for determining the number of blind sources.

• *Temporal FA and Higher order HMM* Both the DFA and Bernoulli FA have been further extended to taking temporal relation among samples in consideration [80, 82] via adding a state equation $y_t = By_{t-1} + \varepsilon_t$ with B being a diagonal and ε_t is a white noise. Specifically, DFA is extended into the so called temporal FA (TFA). Interestingly, as shown in [80, 82], the rotation indeterminacy of DFA has been removed due to temporal relation. While Bernoulli FA is extended into the so called independent hidden Markov model. Moreover, the temporal extensions have also been applied to perform generalized APT financial analyses.

2.5 Bi-directional Mapping: LMSER-ICA and Helmholtz machine

A backward mapping focuses on how x is generated from y such that not only noise is taken in consideration but also which components of y are principal or minor can be evaluated according to their roles in the reconstruction and its matching to the observed data samples. As a result, it makes the problem of selecting an appropriate m and eq.(25) meaningful. However, the disadvantage of a backward mapping is that it is expensive and also inconvenient, based on the learning results of a backward mapping, to perform the mapping $x \to y$. The

disadvantage can be remedied by a bi-directional architecture that combines both a forward mapping and a backward mapping. Actually, the nonlinear LMSER in eq.(15) is a simple example. It combines a forward mapping $y = S(Wx)$ and a backward mapping $x = W^t y$, both are considered during the learning on W. As discussed in Sec.2.2, it performs ICA. Moreover, this ICA has a feature that components can be assessed as being principal or minor by evaluating how good its backward mapping $x = W^t y$ matches the observed data samples, similar to PCA. Thus, the ICA performed by this LMSER is more appropriately to be regarded as a P-ICA. Generally, the reconstruction error $J(W) = E\|x - W^t S(Wx)\|^2$ or even its linear version $J(W) = E\|x - W^t(WW^t)^{-1}Wx\|^2$ can be used to measure $y = WA$ in implementing ICA, e.g., it is P-ICA when the minimum of $J(W)$ is reached or it is M-ICA when the maximum $J(W)$ is reached.

Several extensions of the LMSER learning have been obtained in help of Bayesian Yang-Ying harmony learning [81]. Here we introduce one example, which is equivalent to minimize

$$J(A, W, \{q_r\}_{r=1}^m) = d \ln \sigma + \frac{1}{N}\sum_{t=1}^N \sum_{r=1}^m [y_t^{(r)} \ln q_r + (1 - q_r) \ln (1 - y_t^{(r)})],$$
$$y_t = Wx_t, \quad \sigma^2 = \frac{1}{N}\sum_{t=1}^N \|x_t - AS(y_t)\|^2. \tag{26}$$

This minimization can be implemented by an adaptive EM-like algorithm

$$\begin{aligned} E\ step:\ & y_t = W^{old}x_t,\ \hat{y}_t = arg\min_{y_t}[0.5\sigma^{-2}\|x_t - AS(y_t)\|^2 \\ & -\textstyle\sum_{r=1}^m [y_t^{(r)} \ln q_r + (1 - y_t^{(r)}) \ln (1 - q_r)], \\ M\ step:\ & q_r = \frac{1}{1 + e^{c_r}},\ c_r^{new} = c_r^{old} + \eta(\hat{y}^{(r)} - q_r^{old}),\ e_t = x_t - A^{old}S(\hat{y}_t), \\ & A^{new} = A^{old} + \eta e_t S^T(\hat{y}_t),\ \sigma^{2\ new} = (1 - \eta)\sigma^{2\ old} + \eta d^{-1}\|e_t\|^2 \\ & W^{new} = W^{old} + \eta S_d(\hat{y}_t) e_t x_t^T,\ S_d(y) = diag[s'(y(1)), \cdots, s'(y^{(m)})], \end{aligned} \tag{27}$$

Again, similar to eq.(15), this adaptive harmony learning will push those redundant q_j towards zero such that model selection is automatically made during learning. Alternatively, we also have the model selection criterion

$$\min_m J(m), J(m) = 0.5d \ln \sigma^2 + \textstyle\sum_{j=1}^m [q_r \ln q_r + (1 - q_r) \ln (1 - q_r)]. \tag{28}$$

Furthermore, it has been shown in [80, 82, 83] that this LMSER-ICA eq.(15) is a special case of the Bayesian Yang-Ying independence learning, and from which we also get other extensions of the nonlinear LMSER that not only relax $x = W^t y$ to $x = Ay$ without the constrain $A = W^T$ but also take several possible distributions of y in consideration. Particularly, one special case is equivalent to the one layer deterministic Helmholtz machine learning [31, 16, 17]. Furthermore, extensions have also been made towards to temporal situations, resulting in temporal LMSER [80].

3 Task 2: Mining Groups among Samples

Mining groups or called clustering are made among samples such that samples within a group are regarded as same or similar while samples in different groups

are regarded being different. More specifically, we can classify the studies on the tasks into two categories:

(a) *Homogeneous grouping* Samples are grouped based on a criterion of similarity or distance $d(x_i, x_j)$ that is homogeneous to any sample pair x_i, x_j. A typical example is the Euclidean distance, as further discussed in Sec.3.1.

(b) *Nonhomogeneous grouping* Samples are grouped under a measure that is not homogeneous to every sample, but relates to the specific structure of each group, as further discussed in Sec.3.2.

In particular, when x has a high dimension, which is usually the case in many real applications and especially in multimedia processing, samples are grouped based on each group's specific structure in a subspace of much lowered dimension instead of in the full original space of x. We call such particular cases *Subspace structure based nonhomogeneous grouping*, which will be discussed in Sec.4.

3.1 MSE-VQ Clustering and RPCL Learning

• *MSE Clustering, VQ and KMEAN algorithm* Extensive studies have been made in literature of statistics and pattern recognition for several decades under the name of clustering analysis [4, 21, 35, 19]. The most widely used homogeneous measure is the Euclidean distance $d(x_i, x_j) = \|x_i - x_j\|^2$. The grouping tasks in this situation is equivalent to use a number of vectors to represent a data set such that each vector locates at the center of each group or cluster. The existing algorithms for the purpose can be classified into two types. One is usually called incremental/hierarchical/dynamic clustering [21, 19, 35] or competitive learning [29]. The key point is incrementally adding one cluster center once a newly coming sample is regarded to be far beyond a threshold. This type is easy to implement and the number of clusters is decided dynamically. However, the performances highly depend on the initialization and the specific way that those clusters grows up.

The other type of clustering algorithms considers all the possible cluster centers in parallel via minimizing the mean square distances or a global measure on all the samples. However, the complexity of finding the global minimum grows exponentially with the number of clusters, and thus the problem is usually tackled by a heuristic algorithm that usually produces a solution at a local minimum. A typical example is the KMEAN algorithm [21, 19] and variants, which is also called Vector Quantization (VQ) in literature of speech and image processing [51, 60]. Such a task is equivalently tackled under the name of competitive learning in the literature of neural networks [1, 107]. The KMEAN algorithm has also been adopted and modified in the literature of data mining as one of most popular tools for compressing, categorizing, and organizing data, with emphasis on scaleable ability for a large database [22]. Readers are referred to these cited textbooks, survey papers and references, particularly to a recent survey paper [45] on multimedia data processing.

In this paper, we emphasize certain essential issues in real applications and especially in multimedia data processing. Specifically, two issues will introduce below and several other issues in Sec.3.2.

• *Deciding the number of clusters* The KMEAN algorithm and others work well only when a correct number k of clusters are pre-given. We can get a very poor performance with a wrong data structure found if we do not know this number and thus set k inappropriately [104]. Moreover, on a training set of samples of x, though using a large k may apparently result in a smaller mean square error, it usually generalizes badly on new samples from data, especially in a changing environment. A possible solution to the problem is to choose a best k^* by a selection criterion. Many heuristic criteria have been proposed in the statistic literature [24, 57, 58, 76, 72]. Recently, based the Bayesian Ying-Yang learning, a simple criterion has been also obtained in companion with the KMEAN algorithm as follow [90]:

$$\min_k J(k), J(k) = \ln k + 0.5d \ln E^2_{MSE}, \tag{29}$$

where d is the dimension of x and E^2_{MSE} is the mean square error.

However, any selection criterion suffers a large computational cost since we need to make clustering at a number of different value of k, even though such a process can be organized in a more efficient way, e.g., embedding the evaluation of the selection measure during clustering as did in ISODATA[4]. Alternatively, proposed firstly in 1992 [104], the so called rival penalized competitive learning (RPCL) solves this problem with the correct number k^* determined automatically during learning for an initial k that is large enough, in the sense that extra units are driven far away from data due to appropriately penalized learning on the rival. Later, RPCL has been adopted to various applications, including information retrieval in image databases[48, 41, 46], Plant diagnosis[26], nonlinear financial prediction, and hidden Markov model [13, 12], clustering, vector quautization, object classification in 3-D images, scene segmentation in 2D&3D image as well as multidimensional data [9, 49, 14, 54]. Also, following the initial suggestion in [104] for training RBF net, a number of authors have used or recommended RPCL algorithm for the training of various RBF nets [6, 10, 112, 11, 47, 9]. Subsequently, we will further introduce various RPCL extensions to elliptic clustering and subspace structure based nonhomogeneous grouping as well as its relation to the BYY harmony learning.

• *Fast implementation in a binary tree* In data mining on a large database with data of high dimension, a clustering algorithm that can be fast implemented is preferred. Also, the clusters should be well indexed and thus can be retrieved conveniently. A so called hierarchical PCA [97, 96] for vector quantization provides a solution for such demands. By this technique, a binary clustering tree is formed by recursively splitting a set D_c of samples associated with the current node into two subsets that associate two son nodes such that samples of each part locate on each side of a hyperplane that passes the mean of D_c and is perpendicular to the principal component direction of D_c. After each split, the node

associated with D_c is marked CLOSED. Moreover, we can compute the value of $J(k)$ by the above eq.(29) and check whether $J(k)$ turns to increase. If not, the two sons are marked OPEN. Otherwise, we stop and discard the two sons. Next we go to another OPEN node in either the depth-first or the breadth-first way [67]. The root node of the tree is associated with the entire data set of samples. The tree grows as such until $J(k)$ turns to increase on all the OPEN nodes. Such a technique of forming a data tree can be used for fast indexing and retrieving in multimedia data processing.

3.2 Gaussian Mixture, EM Variants and Eliptic RPCL learning

• *Gaussian Mixture and EM Algorithm* The above discussed algorithms apply to homogeneous data with each being spherical Gaussian and sharing a same or similar portion of samples. Studies in the literatures have also been made on extending the KMEAN algorithm and competitive learning algorithms to clusters of the so called elliptic shapes. Most of these studies can be related to the special cases of the ML learning on Gaussian mixture

$$p(x_t|\theta) = \sum_{j=1}^{k} \alpha_j G(x|m_j, \Sigma_j) \tag{30}$$

in help of the EM algorithm [18, 70, 56] with good convergence properties [92]. E.g., in [90], a simplified EM algorithm on Gaussian mixture is shown to extend the KMEAN algorithm to elliptic clustering. Actually, density estimation by a Gaussian mixture covers the various tasks of clustering with each group represented in a Guassian $G(x|m_j, \Sigma_j)$. It not only provides more accurate clustering results but also sophisticated data structure via Σ_j. As a popular topic, Gaussian mixture with the EM algorithms has been both extensively studied and widely used in many fields. The readers are referred to [45] for its applications in multimedia data processing and to [18, 70, 56] in a broad scope. Here, we add on several results that improve the generalization ability of learning.

• *Three Variants of The EM algorithm* From BYY learning [90, 83, 81], we get as follows:

(a) *Re-weighted EM* We can replace the likelihood $N^{-1}\sum_t \ln p(x_t|\theta)$ by the general convex likelihood $N^{-1}\sum_t f(p(x_t|\theta))$ with a convex function $f'(r) > 0, f''(r) < 0, r > 0$. We maximize this likelihood on a Gaussian mixture $p(x_t|\theta)$ by a so called re-weighted EM algorithm in [90] since a re-weighting factor is attached to each sample, which was shown empirically to be more robust than the ML learning via the original EM algorithm, especially when $f(r) = r^\beta, \beta < 1$.

(b) *Smoothed EM* The performance of the ML learning will degenerate considerably on a set of finite number of high dimensional samples. To solve the problem, a so called smoothed ML learning is proposed, which replaces the likelihood function $N^{-1}\sum_t \ln p(x_t|\theta)$ with an integral $\int p_h(x) \ln p(x_t|\theta)dx$ and $p_h(x)$ given by a Parzen window estimator [83, 81]. Moreover, we are lead to a modified EM algorithm that simply modifies the original EM algorithm at its M-step with its updating on each Σ_j added a smoothing parameter h to its

diagonal elements. Furthermore, after each iteration of the E-step and M-step, we can also update the parameter h via a simple one dimensional search [83, 81].

(c) *De-learning EM* Another special case of BYY harmony learning on Gaussian mixture, also taking the effect of finite number of high dimensional samples in consideration, avoids the smoothing parameter h with its role replaced by a de-learning in the M-step.

To illustrate, we provide a unified adaptive EM-like procedure that covers all the above three algorithms:

$$\begin{aligned} &E\ Step: j_c = arg\max_j[\ln G(x|m_j, \Sigma_j) + \ln \alpha_j], \\ &M\ Step: m_{j_c}^{new} = m_{j_c}^{old} + \eta(x - m_{j_c}^{old}), \\ &\Sigma_{j_c}^{new} = (1-\eta)\Sigma_{j_c}^{old} + \eta[hI + (x - m_{j_c}^{old})(x - m_{j_c}^{old})^T]. \end{aligned} \tag{31}$$

Specifically, it implements the smooth EM with a constant step size $\eta > 0$, together with $h > 0$ updated via a simple one dimensional search [83, 81]. Moreover, when $h = 0$, according to different settings of $\eta > 0$, it acts as:

(1) An elliptic adaptive KMEAN algorithm for a constant step size $\eta > 0$;

(2) An adaptive re-weighted EM algorithm as in [90] for $\eta = f'(p(x_t|\theta))\ p(x_t|\theta)\eta_0 > 0$ with η_0 being a constant;

(3) An adaptive de-learning EM algorithm as in [81] for $\eta = \eta_0\eta_t > 0$, where η_t is in the form of eq.(24) but with $\gamma_t = p(x_t|\theta)$ and $S_q = \sum_t p(x_t|\theta)$. Again, S_q can be approximated by adaptively updating $S_q(t+1) = (1-\lambda)S_q(t) + p(x_t|\theta)$.

• *Selection of Gaussians* Similar to the homogeneous clustering, how to decide the number k of Gaussian is an essential issue for a good performance of Gaussian mixture. Again, one solution is to choose a best k^* by a selection criterion. In [90], such a criterion is obtained as follow:

$$\min_k J(k), J(k) = 0.5\textstyle\sum_{j=1}^k \alpha_j \ln|\Sigma_j| - \sum_{j=1}^k \alpha_j \ln \alpha_j. \tag{32}$$

• *Elliptic RPCL learning and BYY harmony Learning* Also, the correct number k^* of Gaussians can be determined automatically during learning in help of extending RPCL learning [104] to the cases of any elliptic shapes and in any portion of samples [95, 85, 81]. As a result, we have the following elliptic RPCL algorithm:

$$\begin{aligned} &Step\ 1: j_c = arg\max_j d_j(x),\ j_r = arg\max_{j\neq j_c} d_j(x), d_j(x) = -\ln[G(x|m_j, \Sigma_j)\alpha_j], \\ &Step\ 2: m_{j_c}^{new} = m_{j_c}^{old} + \eta_c(x - m_{j_c}^{old}),\ m_{j_r}^{new} = m_{j_r}^{old} - \eta_r(x - m_{j_r}^{old}), \\ &\quad S_{j_c}^{new} = S_{j_c}^{old} + \eta_c \Delta S_{j_c},\ S_{j_r}^{new} = S_{j_r}^{old} - \eta_r \Delta S_{j_r}, \end{aligned} \tag{33}$$

where the learning rate η_c is much smaller than the de-learning rate η_r, e.g., $8 \leq \eta_c/\eta_r \leq 15$. Also, we indirectly compute $\Sigma_j = S_j S_j^T$ via updating S_j in order to guarantee that $\Sigma_{j_c} = S_{j_c} S_{j_c}^T$ remains semi-positive definitive. In eq.(33), ΔS_j is the gradient direction given as follows:

$$\Delta S_j = \nabla_{S_j} \ln G(x|m_j, \Sigma_j) = \{\Sigma_j^{-1}[hI + (x - m_j)(x - m_j)^T]\Sigma_j^{-1} - \Sigma_j^{-1}\}S_j.$$

In implementing, we can always keep S_j in storage or get it at each updating by decomposing Σ_j that is always keep S_j in storage. In the latter case, the updating on S_{j_c} can also be replaced by $\Sigma_{j_c}^{new} = (1-\eta_c)\Sigma_{j_c}^{old} + \eta_c[hI + (x - m_{j_c}^{old})(x - m_{j_c}^{old})^T]$, and eq.(33) degenerates back to eq.(31) when $\eta_r = 0$. Moreover, we can also let η_r to be different for m_j and for Σ_j, even when $\eta_r = 0$ for Σ_j but $\eta_r > 0$ for m_j.

The role of h is same as above discussed. When $h > 0$, we get the smoothed RPCL learning. Also, we can let $\eta_c = f'(p(x_t|\theta))p(x_t|\theta)\eta_{0,c} > 0$ and $\eta_r = f'(p(x_t|\theta))p(x_t|\theta)\eta_{0,r} > 0$ to get the robust feature of the Re-weighted EM algorithm. Though, RPCL is originally proposed heuristically [104], it has been shown that it is qualitatively equivalent to a special case of the general RPCL learning algorithm obtained from the BYY harmony learning [82, 83] and thus get a guide for determining the learning rate η_c and de-learning rate η_r.

4 Task 3: Mining Dependences within Local Groups

We can get the dependence structure among components locally on each cluster in data. One way to do so is anyone of the algorithms in Sec.3.2 for nonhomogeneous grouping to get every covariance matrix Σ_j and then get local dependence structures based on Σ_j. However, since Σ_j contains only the 2nd order statistics, based Σ_j we can not implement local ICA or find a local nonlinear dependence structure. Moreover, even theoretically we can get a linear dependence structure based on Σ_j, not only it wastes many computing costs on getting Σ_j, but also it may result in a bad estimate when the dimension d of x is high, because each Σ_j contains $d(d+1)/2$ parameters to be specified, which needs a large number of samples to avoid the resulted Σ_j to be singular.

A better alternative is to make a nonhomogeneous clustering based on mining local dependence structures that are locally within much lower dimensional subspaces. Specifically, we can get the local extensions of those algorithms in Sec.2.

4.1 Local PCA, Competitive ICA and Modular Models

• *Local PCA and Local PSA* In [109], PCA and PSA are used for local subspace representation of data in pattern recognition. In [100, 101, 96], local PCA is also used for fitting a number of lines and hyperplanes.

Provide that m_j is the center of the j-th cluster, the diagonal elements of the diagonal matrix Λ_j are the d_j largest eigen-values, and the d_j row vectors of W_j are the corresponding eigen-vectors, we define the following subspace based distance

$$d_j(x) = [W_j(x - m_j)]^T \Lambda_j^{-1} W_j(x - m_j), \tag{34}$$

and then use eq.(33) for implementing local PCA with

$$\Lambda_j = S_j S_j^T, \ \Delta S_j = S_j^{-1} diag[(hI + W_j(x - m_j)(x - m_j)^T W_j^T) - \Lambda_j], \tag{35}$$

where S_j is diagonal and $diag[A]$ means a diagonal matrix that takes the diagonal part of A. Moreover, at each location we update as follows

$$Step\ 3: \quad W_{j_c}^{new} = W_{j_c}^{old} + \eta_c \Delta W_{j_c}^{pca},\ W_{j_r}^{new} = W_{j_r}^{old} + \eta_r \Delta W_{j_r}^{mca}, \quad (36)$$

where $\Delta W_{j_c}^{pca}$ can use one existing stable adaptive PCA learning rule on $x'_{j_c} = x - m_{j_c}$ with a linear net $W_{j_c}^{pca} x'_{j_c}$, and $\Delta W_{j_r}^{mca}$ can use one existing stable adaptive MCA learning rule on $x'_{j_r} = x - m_{j_r}$ with a linear net $W_{j_r}^{mca} x'_{j_r}$. Particularly, we can get local PSA as a special case by simply setting $\Lambda_j = \lambda$.

• *Competitive ICA* Instead of exploring local de-correlation structures by Local PCA, we can also explore local independent structures via a so called competitive ICA algorithm:

$$\begin{aligned} &Step\ 1: j_* = arg\max_j [0.5 \ln |W_j W_j^T| + \ln q(W_j x + \mu_j | \theta_y)], \quad (37) \\ &Step\ 2: W_{j_*}^{new} = W_{j_*}^{old} + \eta [I + \phi(y^*)(W_{j_*}^{old} x_t)^T] W_{j_*}^{old}, \\ &\quad \phi(y) = \nabla_y \ln q(y|\theta_y),\ y^* = W_{j_*}^{old} x + \mu_{j_*}^{old}, \\ &\quad \mu_{j_*}^{new} = \mu_{j_*}^{old} + \eta \phi(\hat{y}_t),\ \theta_y^{new} = \theta_y^{old} + \eta \nabla_{\theta_y} \ln q(W_j x + \mu_j | \theta_y). \end{aligned}$$

For a fixed j, Step 2 is the same as the learning parametric mixture based ICA [88, 86], where a finite mixture is used as $q(y^{(j)}|\theta_y^{(j)})$, with θ_y updated by an EM-like algorithm during updating W_j. Moreover, we can also generalize it to competitive temporal ICA for handling temporal situation [80].

• *Modular Supervised Learning: RBF net, mixture-of-experts, and support vectors* Local dependence structures can also be built via supervised learning on modular models such as the radial basis function (RBF) [59, 61, 102], nonparametric kernel regression [20, 102], the mixture-of-expert (ME) models [34, 37, 38], and support vector machine [79]. Similar to the previously discussed unsupervised learning examples, these supervised modular models also build dependence structures among components based on local properties of data. E.g., the conventional learning on RBF nets is made usually in two sequential steps. The first step decides the centers of basis functions usually via certain clustering algorithm, and the second step determines the parameters of the output layers by the least square learning. Such a two-step algorithm actually provides a suboptimal solution. Extensive literatures are available on these supervised learning models. For a more detailed introduction, readers are referred to [108] for an early survey and to [81] for a recent discussion on the relation between these models.

Here we summarize several results on training these models by either adaptive EM-like algorithms or RPCL related algorithms:

(a) The mixture-of-expert (ME) model [34, 37, 38] implements forward mapping by a number of local experts that are engaged in via a probabilistic controlling of a so called gating net, with each individual expert being a three layer net. Moreover, an alternative ME model is further proposed [98, 69, 84] such that learning can be made completely by the EM algorithm in the case that each expert is described by a Gaussian with a linear regression, while the training on

the gating net of the original ME is trained by a gradient-based algorithm but not by the EM algorithm.

(b) The normalized RBF nets and the extended normalized RBF nets are shown in [84] to be regarded as special cases of the alternative mixture-of-expert (ME) model, and thus can be trained by ML learning by the EM algorithm, instead of the conventional two-step method. Moreover, the hard-cut EM algorithm and adaptive EM-like algorithms have been proposed for fast learning on not only these RBF nets but also both the original and alternative ME models in help of a so called coordinated competition [84].

(c) In [82, 81], all the above studies are be related to the Bayesian Ying-Yang harmony learning as special cases. As a result, their learning algorithms can be replaced by their corresponding RPCL-type learning algorithms that perform parameter learning with automated model selection on experts or basis functions. Also, criteria are obtained in a way similar to eq.(19).

(d) In [102], nonparametric kernel regression [20] is shown to be a special case of the normalized RBF nets such that several previous results on kernel regression can be brought to provide certain understandings on the normalized RBF nets. Recently in [81], such a link is revisited from the perspective of using the above discussed learning algorithms on generalizing kernel regression technique, resulting in not only an easily implemented approach for determining the smoothing parameter in kernel regression, but also an alternative approach to select supporting vectors in the popular supporting vector machines for a better generalization.

4.2 Local MCA-MSA and Curve Detection

As a dual to local PCA-PSA discussed in Sec.4.1, local MCA-MSA can be used for local subspace representation of data [109], for fitting curve, hyperplane and hypersurface [107] and detecting a number of curves, hyperplanes and hypersurfaces at different locations [104, 100, 101, 96].

Moreover, as a dual to the RPCL algorithm in Sec.4.1, local MCA-MSA can also be implemented by RPCL learning. Specifically, we can replace eq.(34) by

$$d_j(x) = \|(I - W_j^T W_j)(x - m_j)\|^2, \tag{38}$$

and then use eq.(33) with eq.(35) for learning. The difference is that eq.(36) is replaced by

$$Step\ 3: W_{j_c}^{new} = W_{j_c}^{old} + \eta_c \Delta W_{j_c}^{mca},\ W_{j_r}^{new} = W_{j_r}^{old} + \eta_r \Delta W_{j_r}^{pca}, \tag{39}$$

where the positions of using a PCA rule and a MCA rule are swapped.

Taking curve detection as an example, this technique provides an alternative to the Hough transform-like technique on detecting curves on image in noisy environment [111, 105]. Such tasks may be implemented in two ways. One is to use the trick in [107] to transfer a curve into a form such that the above local MCA can be used directly. E.g., for detecting a quadratic curve such as circles or ellipses, we consider the equation $a_j x^2 + b_j xy + c_j y^2 + d_j x + e_j y +$

$f_j = 0$ and rewrite it into $w_j^T(x - m_j) = 0$ with $x = [x^2, xy, y^2, x, y]^T$ and $w_j = [a_j, b_j, c_j, d_j, e_j]^T$, and then. Therefore, we can perform a local MCA by the above algorithm at a special case that each W_j consists of only one vector of w_j. After learning, we turn each resulted w_j into parameters of curve with $f_j = -w_j^T m_j$.

Another way is to define $d(x, \theta_j)$ as the shortest distance from x to the j-th curve represented by θ_j, and then use RPCL learning as follows

$$\begin{aligned} &Step\ 1: j_c = arg\max_j d(x, \theta_j),\ j_r = arg\max_{j \neq j_c} d(x, \theta_j), \\ &Step\ 2: \theta_{j_c}^{new} = \theta_{j_c}^{old} + \eta_c \nabla_{\theta_{j_c}} d_j(x, \theta_{j_c}),\ \theta_{j_r}^{new} = \theta_{j_r}^{old} - \eta_r \nabla_{\theta_{j_r}} d_j(x, \theta_{j_r}). \end{aligned} \tag{40}$$

More generally, in the so called Multi-sets modeling [95], $d(x, \theta_j)$ can be the shortest distance from x to the j-th object described in a general set, and we use eq.(40) for learning.

4.3 Local Backward Mapping and Competitive LMSER

• *Backward mapping: Local DFA and Local Independent FA* The advantage of using a Gaussian mixture for mining groups and the advantage of using the factor model eq.(20) for mining the dependence structure can be combined. We consider the following two possibilities:

$$p(x_t|\theta) = \sum_{j=1}^{k} \alpha_j \begin{cases} \int G(x|A_j y + m_j, \Sigma_j) G(y|0, I) dy, & \text{(a)}, \\ \int G(x|Ay, \Sigma) G(y|\mu_j, \Lambda_j) dy, & \text{(b)}. \end{cases} \tag{41}$$

In the case (a), at each location m_j, each Gaussian is decomposed into a local DFA model

$$x = m_j + A_j y_j + e_j, \tag{42}$$

where e_j is a Gaussian noise of zero mean and covariance matrix Σ_j. Thus, similar to a DFA previously discussed in Sec.2.4, there is still indeterminacy on rotation and scale at every location. This situation is removed by in the case (b) in eq.(41) where a Gaussian mixture factor $\sum_{j=1}^{k} \alpha_j G(y|\mu_j, \Pi_j)$ is mapped to x via a common $x = Ay + e$ or equivalently each local DFA locates at $A\mu_j$. Due to the constraint of this common mapping, the indeterminacy on rotation at every location is removed except the singular case that the distribution is same at each location.

Moreover, the local structure based Gaussian mixture in eq.(41) can be further extended to nonGaussian finite mixture $p(x|\theta) = \sum_{j=1}^{k} \alpha_j p(x|\theta_j)$ by letting $G(y_j|0, I)$ or $G(y|\mu_j, \Lambda_j)$ replaced by the independent factor eq.(1), which leads to two corresponding local independent FA models.

Furthermore, we can combine eq.(31) and eq.(21) to get a double loop EM-like adaptive algorithm for implementing learning on local DFA and independent

FA. Taking the case (a) as an example, we have

$$
\begin{aligned}
&E\ \ Step: j_c = arg\max_j[\ln p(x|\theta_j) + \ln \alpha_j],\\
&\quad p(x|\theta_j) = \begin{cases} \int G(x|A_j y_j + m_j, \Sigma_j) G(y_j|0, I) dy_j, & \text{Gaussian,} \\ \int G(x|A_j y_j + m_j, \Sigma_j) q(y|\theta_{y,j}) dy, & q(y|\theta_{y,j}) \text{ given by eq.(1),} \end{cases}\\
&M\ \ Step: \text{ implement the inner E step and M step in eq.(21) once, with}\\
&\quad A_{j_c}\ as\ A,\ \ x - m_{j_c}\ as\ x,\ \ \Sigma_{j_c}\ as\ \Sigma_e,\ \ \theta_{y,j}\ as\ \theta_y,\\
&\quad Then,\ update\ m_{j_c}^{new} = m_{j_c}^{old} + \eta(x - m_{j_c}^{old}).
\end{aligned} \tag{43}
$$

• *Bi-directional Mapping: Competitive LMSER* We can extend the LMSER learning eq.(26) to the local models at different m_j. Similar to eq.(43), we can get the following double loop EM-like adaptive algorithm for learning, with its E step in the outer loop implementing competition each time a sample comes:

$$
\begin{aligned}
&E\ \ Step: j_c = arg\max_j J(A_j, W_j, \{q_{j,r}\}),\ \ y_t = W_j(x_t - m_j),\\
&\quad J(A_j, W_j, \{q_{j,r}\}) = 0.5d \ln \sigma_j^2 + 0.5\sigma_j^{-2}\|x_t - A_j S(y_t)\|^2\\
&\quad -\textstyle\sum_{r=1}^{d_j}[y_t^{(r)} \ln q_{j,r} + (1 - q_{j,r}) \ln (1 - y_t^{(r)})],\\
&M\ \ Step: \text{ implement the inner E step and M step in eq.(27) once, with}\\
&\quad A_{j_c}\ as\ A,\ \ W_{j_c}\ as\ W,\ \ x_t - m_{j_c}\ as\ x_t,\ \ \sigma_{j_c}^2\ as\ \sigma^2,\ \ q_{j,r}\ as\ q_r,\\
&\quad Then,\ update\ m_{j_c}^{new} = m_{j_c}^{old} + \eta(x - m_{j_c}^{old}).
\end{aligned} \tag{44}
$$

5 Tasks 4 & 5: Topologically Organized Groups and Local Dependence Structures

Another important data structure consists of topological relations among groups or clusters. Early efforts can also be traced back several decades. Roughly, these studies originated along two lines. One is for reducing high dimensional data into lower dimension such that topological relations among samples can be reserved for statistic data analysis and engineering purpose. Several heuristic techniques were proposed. Among them, a typical representative is called Samon mapping [73]. The other line is motivated by mathematical modeling biological striate cortex. A typical work is Malsburg's self-organization of orientation sensitive cells [52]. A breakthrough advance, that is able both to model self-organized formation of topological feature maps in biological system and to apply to data analysis with efficient computing, is the well known Kohonen map [43, 42].

In the past decade, several efforts have been further made in the literature of neural networks for implementing Samon-type mapping in help of nonlinear architecture of neural networks [66, 53, 71]. Moreover, very extensive studies on topological map have been made on Kohonen map, which actually forms a major stream of unsupervised learning in the literature of neural networks, as shown by the main theme of this series of workshops. Readers are referred to papers in the workshops' proceedings and other vast volumes on this theme in the neural network literature.

Here, we only incompletely mention three lines of developments that relate to the local dependence mining methods discussed in Sec.4. One is extending the lattice map architecture to more sophisticated architectures. An early attempt is made in 1990 for training a number of Kohonen maps that are automatically organized in a pipe-line architecture during learning [110]. Further developments along this direction include Kohonen map tree [44] and the Growing Grid or gas [25]. The another line is to combine the feature of Kohonen map with Gaussian mixture such as given in [7]. Another line that deserves to mention is applying certain advanced mathematical results on deformation analyses to interpret and evolve self-organization map [50].

It may deserve to mention that few efforts has been made in the existing literature on combining the tasks of mining local dependence structures into the formation of topological structure yet, which should be a promising direction of developments of studies on self-organizing map.

References

1. Ahalt, S.C., et al, "Competitive learning algorithms for vector quantization", *Neural Networks*, **3**, 277-291, 1990.
2. Amari, S.-I., Cichocki, A., & Yang, H.H., "A new learning algorithm for blind separation of sources", in D. S. Touretzky, et al, eds, *Advances in Neural Information Processing 8,* MIT Press, 757-763, 1996.
3. Anderson, T.W., & Rubin, H., "Statistical inference in factor analysis", *Proc. Berkeley Symp. Math. Statist. Prob. 3rd 5*, UC Berkeley, 111-150, 1956.
4. Ball, G.H., & Hall, D.J., "ISODATA: A novel method of data analysis and pattern classification', *Tech. Rep. No. AD 699616*, Stanford Research International, 1965.
5. Bell, A.J. & Sejnowski, T.J., "An information-maximization approach to blind separation and blind de-convolution", *Neural Computation 7*, 1129-1159, 1995.
6. Billings, S. A., & Zheng, G. L., "Radial basis function network configuration using genetic algorithms", *Neural Networks 8*, 877-890, 1995.
7. Bishop, C.M., Svensen, M. and Williams, C.K.I., " GTM: the Generative Topographic Mapping", *Neural Computation 10*, 215-234, 1998.
8. Bradley, P.S., et al, "Data mining: overview and optimization opportunities", *Microsoft Research Technical Report*, TR-98-04, 1998.
9. Bors, A. G. & Pitas, I., "Object classification in 3-D images using alpha-trimmed mean radial basis function network," *IEEE Trans. on Image Process 8*, 1744-1756, 1999.
10. Bors, A. G. & Pitas, I., "Median radial basis function neural network", *IEEE Trans. on Neural Networks 7*, 1351-1364, 1996.
11. Chang, P. R. & Yang, W. H., "Environment-adaptation mobile radio propagation prediction using radial basis function neural networks", *IEEE Trans. on Vehicular Technology 46*, 155-160, 1997.
12. Cheung, Y. M. & Xu, L., "A RPCL-based approach for Markov model identification with unknown state number," *IEEE Signal Processing Letters 7*, 284-287, 2000.
13. Cheung, Y. M., et al, "A RPCL-CLP architecture for financial time series forecasting," in *Proc. of 1995 IEEE ICNN 2*, 829-832, 1995.
14. Chiarantoni, E., et al, "Scene segmentation in video sequences by a RPCL neural network," *Proc. of 1998 IEEE WCCI 3*, 1877-1882, 1998.

15. Cichocki, A., Douglas, S.C., & Amari, S., " Robust techniques for independent component analysis (ICA) with noisy data", *Neurocomputing 22*, 113-129, 1998.
16. Dayan, P., & Zemel, R.S., "Competition and multiple cause models", *Neural Computation 7*, 565-579, 1995.
17. Dayan, P. & Hinton, G., E., "Varieties of Helmholtz machine", *Neural Networks 9*, 1385-1403, 1996.
18. Dempster, A.P., et al, "Maximum- likelihood from incomplete data via the EM algorithm", *J. of Royal Statistical Society B39*, 1-38, 1977.
19. Devijver, P. A. & Kittler, J., *Pattern Recognition: A Statistical Approach*, Prentice-Hall, 1982.
20. Devroye, L., et al, *A Probability Theory of Pattern Recognition*, Springer, 1996.
21. Duda, R.O., & Hart, P.E., *Pattern classification and Scene analysis*, Wiley, 1973.
22. Fayyad, U.M., et al " Knowledge discovery and data mining: towards a unifying framework", *Proc. 2nd Intl. Conf. KDD-96*, 1996.
23. Fisher, R.A., "The Use of multiple measurements in taxonomic problems", *Ann. Eugen 7*, 178-188, 1936.
24. Friedman, H.P., & Rubin, J., " On Some invariant criteria for grouping data", *J. Amer. Statis., Assoc. 62*, 1159-1178, 1967.
25. Fritzke, B, "Growing Grid- A self-organizing network with constant neighborhood range and adaptation strength", *Neural Processing Letters 2*, (5), 1995.
26. Furukawa, H., et al, "A systematic method for rational definition of plant diagnostic symptoms by self-organizing neural networks", *Neurocomputing 13*, 171-183, 1996.
27. Fyfe, C., ed., Special issue on *Independence and artificial neural networks*, *Neurocomputing 22*, No.1-3, 1998.
28. Glymour, C. et al, " Statistical Themes and lessons for data mining", *Data Mining and Knowledge Discovery 1* , 25-42, 1996.
29. Grossberg, S., "Competitive learning: from interactive activation to adaptive resonance", *Cognitive Science 11*, 23-63, 1987.
30. Hamilton, J.D., *Time Series Analysis*, Princeton University Press, New Jersy, 1994.
31. Hinton, G. E., et al, "The wake-sleep algorithm for unsupervised learning neural networks", *Science 268*, 1158-1160, 1995.
32. Hotelling. H, "Simplified calculation of principal components", *Psychometrika 1*, 27-35, 1936.
33. Hyvarinen, A. "Survey on Independent Component Analysis", *Neural Computing Surveys 2* , 94-128, 1999.
34. Jacobs, R.A., et al, "Adaptive mixtures of local experts", *Neural Computation 3*, 79-87, 1991.
35. Jain, A.K., & Dubes, R.C., *Algorithm for Clustering Data*, Prentice-Hall, 1988.
36. Jensen, F.V., *An introduction to Bayesian networks*, University of Collage London Press, 1996.
37. Jordan, M. I., & Jacobs, R.A., " Hierarchical mixtures of experts and the EM algorithm", *Neural Computation 6*, 181-214, 1994.
38. Jordan, M. I., & Xu, L., "Convergence results for the EM approach to mixtures of experts", *Neural Networks 8*, 1409-1431, 1995.
39. Jutten, C. & Herault, J., " Independent Component Analysis versus Principal Component Analysis", *Proc. European Signal Processing Conf EUSIPCO88*, 643-646, 1988.
40. Karhunen, J. & Joutsensalo, J., "Representation and separation of signals using nonlinear PCA type Learning," *Neural Networks 7*, 113-127, 1994.

41. King, I., et al, "Using rival penalized competitive clustering for feature indexing in Hong Kong's textile and fashion image database," *Proc. of 1998 IEEE WCCI 1*, 237-240, 1998.
42. Kohonen, T, *Self-Organizing Maps* , Springer-Verlag, Berlin, 1995.
43. Kohonen, T., "Self-organized formation of topologically correct feature maps", *Biological Cybernetics 43*, 59-69, 1982.
44. Koikalainen, P, " Progress with the Tree structure Self-Organizing Map", Proc. ECAI94: 11th European Conference on Artificial Intelligence, Ed. A.Cohn, John Wiley&Son, 211-215, 1994.
45. Kung, S. Y., & Hwang, J.N., "Neural Networks for Intelligent Multimedia Processing," *Proceedings of the IEEE, 86*, No. 6, 1244-1272, 1998.
46. Lau, T.K. & King, I., "Performance analysis of clustering algorithms for information retrieval in image databases," *Proc. of 1998 IEEE WCCI 2*, 932-937, 1998.
47. Lee, J., et al, "A practical radial basis function equalizer," *IEEE Trans. on Neural Networks 10*, 450-455, 1999.
48. Li, X. Q., & King, I., "Regression analysis for rival penalized competitive learning binary tree," *Proc. of IJCNN2000 6*, 290-295, 2000.
49. Li, R., et al, "Fast image vector quantization using a modified competitive learning neural network approach" *Intl J. of Imaging Systems and Technology 8*, 413-418, 1997.
50. Liou, C.Y. and Tai, W.P., "Conformality in the self-organization network", *Artificial Intelligence, 116*, 265-286, 2000.
51. Makhoul, J., Rpucos, S., & Gish, H., "Vector quantization in speech coding," *Proc. IEEE*, **73**, 1551-1558, 1985.
52. von der Malsburg, Ch. "Self-organization of orientation sensitive cells in the striate cortex", *Kybernetik 14*, 85-100, 1973.
53. Mao, J, and Jain, A.K., " Artificial neural networks for feature extraction and multivariate data projection", *IEEE Trans. Neural Networks 6*, No. 2, 296-317, 1995.
54. Marazzi, A., et al, "Automatic selection of the number of clusters in multidimensional data problems", *Proc. of Intl. Conf. on Image Processing 3*, 631-634, 1996.
55. McDonald, R, *Factor Analysis and Related Techniques*, Lawrence Erlbaum, 1985.
56. McLachlan, G.J., & Basford, K.E., *Mixture Models: Inference and Application to Clustering*, Dekker, 1988.
57. Millgan, G. W., " A monte carlo study of thirty internal criterion measures for cluster analysis", *PSYCHOMETRIKA 46*, 187-199, 1985.
58. Millgan, G. W., & Copper, M.C., " An examination of procedures for determining the number of clusters in a data set", *PSYCHOMETRIKA 50*, 159-179, 1985.
59. Moody, J. & Darken, J., "Fast learning in networks of locally-tuned processing units", *Neural Computation 1*, 281-294, 1989.
60. Nasrabadi, N. & King, R. A., "Image coding using vector quantization: a review," *IEEE Trans. Commun. 36*, 957-971, 1988.
61. Poggio, T., & Girosi, F., "Networks for approximation and learning", *Proc. of IEEE 78*, 1481-1497, 1990.
62. Oja, E., "A simplified neuron model as a principal component analyzer", *J. Mathematical Biology, 16*, 267-273, 1982.
63. Oja, E., "Neural networks, principal components, and subspaces", *Int. J. Neural Systems 1*, 61-68, 1989.
64. Oja, E., et al, "Learning in nonlinear constrained Hebbian networks" , *Proc. ICANN'91*, Helsinki, 385-390, 1991.

65. Oja E. & Hyvarinen,A., "Blind Signal Separation by Neural Networks", *Proc. ICONIP96*, Springer-Verlag, 7-14, 1996.
66. Ornes, G and Sklansky, J, " A neural network that visualizes what it classifies", *Pattern Recognition Letters 18*, No.11-13, 1307-1316, 1997.
67. Pearl J. *Heuristics: Intelligent Search Strategies for Computer Problem Solving*, Addison-Wesley, 1984.
68. Pearl, J, *Probabilistic reasoning in intelligent systems: networks of plausible inference*, San Fransisca, CA: Morgan Kaufman, 1988.
69. Ramamurti, V., & Ghosh, J., "Regularization and error bars for the mixture of experts network", *Proc. of IEEE ICNN97*, 221-225, 1997.
70. Redner, R.A., & Walker, H.F., "Mixture densities, maximum likelihood, and the EM algorithm", *SIAM Review 26*, 195-239, 1984.
71. de Ridder, D., and Duin, R.P.W., " Sammnon's mapping using neural networks: a comparision", *Pattern Recognition Letters 18*, No.11-13, 1301-1306, 1997.
72. Rissanen, J. et al, "Unsupervised classification with stochastic complexity in multivariate statistical modeling", *Proc. The 1st US/Japan Conf. on the Frontiers of Statistical Modeling: An In formational Approach*, 1994.
73. Samon, John W., Jr.. "A Nonlinear Mapping for Data Structure Analysis," *IEEE Trans. on Computers, Vol.C-18*, No.5, 1969.
74. Sanger, T.D., " Optimal unsupervised learning in a single-layer linear feed forward neural network", *Neural Networks 2*, 459-473, 1989.
75. Saund, E., "A multiple cause mixture model for unsupervised learning", *Neural Computation 7*, 51-71, 1995.
76. Scott, A.J., & Symons, M.J., "Clustering methods based on likelihood ratio criteria", *Biometrics 27*, 387-397, 1971.
77. Spearman, C., "General intelligence objectively determined and measured", *Am. J. Psychol. 15*, 201-293, 1904.
78. Taleb, A.. & Jutten, C., "Non-linearity source separation: the post-nonlinear mixtures", *Proc.ESANN97*, Bruges, April 16-18, 279-284 , 1997.
79. Vapnik, V.N., *The Nature Of Statistical Learning Theory*, Springer-Verlag, 1995.
80. Xu, L., " BYY Harmony Learning, Independent State Space and Generalized APT Financial Analyses ", in press, *IEEE Trans on Neural Networks*, 2001.
81. Xu, L., "Best Harmony, Unified RPCL and Automated Model Selection for Unsupervised and Supervised Learning on Gaussian Mixtures, Three-Layer Nets and ME-RBF-SVM Models", in press, a special issue on *International Journal of Neural Systems*, 2001. A part of its preliminary version on *Proc. IJCNN99, Vol.1: 540-545* and on *Proc. WCNN96: 193-200.*
82. Xu, L., "Temporal BYY Learning for State Space Approach, Hidden Markov Model and Blind Source Separation", *IEEE Trans on Signal Processing 48*, 2132-2144, 2000. A part of its preliminary version on *Proc. IJCNN99, Vol.2: 949-954* and on *Proc. ICONIP98-Kitakyushu, Vol.2:877-884.*
83. Xu, L., "BYY System and Theory for Statistical Learning: Best Harmony, Data Smoothing, and Model Selection", *Neural, Parallel and Scientific Computations 8*, 55-82, 2000. Its preliminary version on *Proc. 1999 Chinese Conf. on NNSP*, 12-29, 1999.
84. Xu, L., "RBF Nets, Mixture Experts, and Bayesian Ying-Yang Learning", *Neurocomputing 19*, No.1-3, 223-257, 1998.
85. Xu, L., "Rival penalized competitive learning, finite mixture, and multisets clustering," *Proc. of 1998 IEEE IJCNN 3*, 251-2530, 1988.
86. Xu, L., "Bayesian Kullback Ying-Yang Dependence Reduction Theory", *Neurocomputing 22*, No.1-3, 81-112, 1998.

87. Xu, L., and Leung, W.M., "Cointegration by MCA and modular MCA", *Proc. IEEE/IAFE 1998 Intl. Conf. on Computational Intelligence for Financial Engineering (CIFEr)*, March 29-31, NY City, 157-160, 1998.
88. Xu, L., Cheung, C.C., & Amari, S.-I., "Learned Parametric Mixture Based ICA Algorithm", *Neurocomputing 22*, No.1-3, 69-80, 1998. A part of its preliminary version on *Proc. ESANN97*, Bruges, April 16-18, 291-296, 1997.
89. Xu, L., "Bayesian Ying-Yang Learning Based ICA Models" , *Proc. 1997 IEEE Signal Processing Society Workshop*, Sept. 24-26, Florida, 476-485, 1997.
90. Xu, L., "Bayesian Ying-Yang Machine, Clustering and Number of Clusters", *Pattern Recognition Letters* **18**, *1167-1178, 1997.*
91. *Xu, L., Yang, H.H., & Amari,S.-I., "Signal Source Separation by Mixtures Accumulative Distribution Functions or Mixture of Bell-Shape Density Distribution Functions ", Research Proposal, presented at* FRONTIER FORUM (speakers: D. Sherrington, S. Tanaka, L.Xu & J. F. Cardoso), *organized by S.Amari, S.Tanaka & A.Cichocki, RIKEN, Japan, April 10, 1996.*
92. *Xu, L., & Jordan, M. I., "On Convergence Properties of the EM Algorithm for Gaussian Mixtures", Neural Computation 8, 129-151, 1996.*
93. *Xu, L., "A unified learning scheme: Bayesian-Kullback YING-YANG machine",* Advances in NIPS 8, *eds., D. S. Touretzky, et al, MIT Press, 1996, 444-450. A part of its preliminary version on* Proc. ICONIP'95, *1995, 977-988.*
94. *Xu, L., & Yuille, L., "Robust Principal Component Analysis by Self-Organizing Rules Based on Statistical Physics Approach", IEEE Tr. Neural Networks 6, 131-143, 1995.*
95. *Xu, L., "A Unified Learning Framework: Multisets Modeling Learning", Proc. of 1995 World Congress on Neural Networks 1*, 35-42, 1995. A part of its preliminary version on *Proc. 1994 IEEE Intl. Conf. on Neural Networks 1*, 315-320, 1994.
96. Xu, L., "Advances on three streams of PCA studies" *Proc. 1995 IEEE Intl Conf. on NNSP'95*, 480-483, 1995.
97. Xu, L., "Vector Quantization by Local and Hierarchical LMSER", *Proc. 1995 Intl Conf. on Artificial Neural Networks II,* Paris, 575-579, 1995.
98. Xu, L., Jordan, M.I., & Hinton, G.E., "An Alternative Model for Mixtures of Experts", *Advances in Neural Information Processing Systems 7*, 633-640, 1995. Its preliminary version on *Proc. of WCNN'94 2*, 405-410, 1994.
99. Xu, Y. & Xu, L., " An Approach to Missing Data Principal Component Analysis", *Proc. ICONIP'95 II,* 1013-1016, 1995.
100. Xu, L., " Beyond PCA Learning: From Linear to Nonlinear and From Global Representation to Local Representation ", *Proc. ICONIP'94*, 943-949, 1994
101. Xu, L., " Theories for Unsupervised Learning: PCA and Its Nonlinear Extensions", *Proc. IEEE ICNN'94 II,* 1252-1257, 1994.
102. Xu, L., Krzyzak, A., & Yuille, A.L., "On Radial Basis Function Nets and Kernel Regression: Statistical Consistency, Convergence Rates and Receptive Field Size", *Neural Networks 7*, 609-628, 1994.
103. Xu, L., "Least mean square error reconstruction for self-organizing neural-nets", *Neural Networks 6*, 627-648, 1993. Its early version on *Proc. IJCNN91'Singapore*, 2363-2373, 1991.
104. Xu, L., Krzyzak, A., & Oja, E., " Rival Penalized Competitive Learning for Clustering Analysis, RBF net and Curve Detection", *IEEE Trans. on Neural Networks 4*, 636-649, 1993. Its early version on *Proc. of 11th Intl. Conf. on Pattern Recognition 1*, 672-675, 1992.

105. Xu, L, & Oja, E., "Randomized Hough Transform (RHT): Basic Mechanisms, Algorithms and Complexities", *Computer Vision, Graphics, and Image Processing : Image Understanding 57*, No.2, 131-154, 1993.
106. Xu, L., & Klasa, S. (1993), " A PCA Like Rule for Pattern Classification Based on Attributed Graph",*Proc. IJCNN93*, 1281-1284, 1993.
107. Xu, L, Oja, E. & Suen, C. Y., "Modified Hebbian learning for curve and surface fitting", *Neural Networks 5*, 441-457, 1992.
108. Xu, L., Klasa, S. & Yuille, L., "Recent Advances on Techniques of Static Feed-forward Networks With Supervised Learning", *Int. J. Neural Systems 3*, 253-290, 1992.
109. Xu, L, Krzyzak, A., and Oja, E., "A Neural Net for Dual Subspace Pattern Recognition Methods", *Intl. J. Neural Systems, 2*, No.3, 169-184, 1991.
110. Xu, L., "Adding Learned Expectation into The Learning Procedure of Self-Organizing Maps", *Intl. J. Neural Systems, 1*, No.3, 269-283, 1990.
111. Xu, L, Kultanen. P., & Oja, E., "A New Curve Detection Method: Randomized Hough Transform (RHT)", *Pattern Recognition Letters 11*, 331-338, 1990.
112. Zheng, G. L., & Billings, S. A., "Radial basis function network configuration using mutual information and the orthogonal least squares algorithm", *Neural Networks 9*, 1619-1637, 1996.