Radial Basis Function Networks (RBFNs)

- Structure of RBFNs

. Three layers: input, hidden, and output layers

. The hidden layer is composed of kernel functions.

. The output of network is described by

$$\phi(\underline{x}) = \sum_{i=1}^{m} w_i \psi(\underline{x}, \underline{p}_i)$$

where $\underline{p}_i$ represents the parameter vector for the ith kernel function.

- Expressive Capabilities of RBFNs

. Any bounded continuous function $f(\underline{x})$ can be approximated by $\phi(\underline{x})$ provided that

$$\int_{-\infty}^{+\infty} (\frac{1}{a})^n \psi(\underline{x}, \underline{p}_a) d\underline{x} = C(>0) \quad \text{and}$$

$$\lim_{a \to \infty} (\frac{1}{a})^n \psi(\underline{x}, \underline{p}_a) = C\delta(\underline{x})$$

where a represents the kernel width and $\underline{p}_a$ represents the shape parameter. (Lee and Kil, 1991)

That is, for any $\epsilon > 0$, there exists $\phi(\underline{x})$ such that

$|f(\underline{x}) - \phi(\underline{x})| < \epsilon$.

. Examples of kernel functions

$$\psi(x,a) = e^{-x^2/a^2}$$

$$\psi(x,a) = Sinc(\frac{x}{a}) = \frac{a}{\pi x}\sin(\frac{\pi x}{a})$$

$$\psi(x,a) = Rect(\frac{x}{a}) = \begin{cases} 1 & \text{if } |x| < a/2 \\ 0 & otherwise \end{cases}$$

- Learning algorithms

A. Parzen window approach
(of estimating probability density functions)

. The volume of d-dimensional hypercube:

$$V_n = h_n^d$$

where h_n is an edge of hypercube.

. An window function is defined by

$$\psi(\underline{x}) = \begin{cases} 1 & \text{if } |x_i| \leq 1/2,\ i = 1, \cdots, d \\ 0 & otherwise \end{cases}$$

a unit hypercube centered at the origin.

. an estimate of density function:

$$p_n(\underline{x}) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{V_n}\psi(\frac{\underline{x}-\underline{x}_i}{h_n}) \quad \text{...} \quad \text{(1)}$$

where n represents the number of samples and $\underline{x}_i$ represents the ith sample. This estimate converges to $p(\underline{x})$ if

$$\lim_{n\to\infty} E[p_n(\underline{x})] = p(\underline{x}) \quad \text{and} \quad \lim_{n\to\infty} Var(p_n(\underline{x})) = 0.$$

. From the convergence condition, we can deduce the following conditions of (1):

$$\max_{\underline{x}}\psi(\underline{x}) < \infty, \quad \lim_{\|\underline{x}\|\to\infty}\psi(\underline{x})\prod_{i=1}^{d}x_i = 0,$$

$$\lim_{n\to\infty} V_n = 0 \quad \text{and} \quad \lim_{n\to\infty} n V_n = \infty.$$

That is, V_n should be changed as follows:

$$V_n = \frac{V_1}{\sqrt{n}} \quad \text{or} \quad V_n = \frac{V_1}{\log n}. \quad \text{...} \quad \text{(2)}$$

. The condition of $\lim_{n\to\infty} E[p_n(\underline{x})] = p(\underline{x})$:

$$\begin{aligned}\lim_{n\to\infty} E[p_n(\underline{x})] &= \lim_{n\to\infty}\frac{1}{n}\sum_{i=1}^{n}E[\frac{1}{V_n}\psi(\frac{\underline{x}-\underline{x}_i}{h_n})] \\ &= \lim_{n\to\infty}\int\frac{1}{V_n}\psi(\frac{\underline{x}-\underline{v}}{h_n})p(\underline{v})d\underline{v} \\ &= \int\delta(\underline{x}-\underline{v})p(\underline{v})d\underline{v} \\ &= p(\underline{x})\end{aligned}$$

. The condition of $\lim_{n\to\infty} Var(p_n(\underline{x})) = 0$:

$$
\begin{aligned}
Var(p_n(\underline{x})) &= E[p_n^2(\underline{x})] - E^2[p_n(\underline{x})] \\
&= E[(\sum_{i=1}^{n} \frac{1}{n V_n} \psi(\frac{\underline{x} - \underline{x}_i}{h_n}))^2] - E^2[p_n(\underline{x})] \\
&= \sum_{i=1}^{n} E[\frac{1}{n^2 V_n^2} \psi^2(\frac{\underline{x} - \underline{x}_i}{h_n})] - E^2[p_n(\underline{x})] \\
&= \frac{1}{n V_n} \int \frac{1}{V_n} \psi^2(\frac{\underline{x} - \underline{v}}{h_n}) p(\underline{v}) d\underline{v} - E^2[p_n(\underline{x})] \\
&\leqq \frac{\max(\psi) E[p_n(\underline{x})]}{n V_n}
\end{aligned}
$$

Therefore, if V_n satisfies the condition of (2),

$\lim_{n\to\infty} Var(p_n(\underline{x})) = 0$.

B. Learning strategies of estimating continuos functions

Case 1: fixed centers selected at random (Lowe, 1989)

Estimation function is represented by

$$\phi(\underline{x}) = \sum_{i=1}^{m} w_i \psi_i(\underline{x})$$

where

$\psi_i(\underline{x}) = e^{-m \| \underline{x} - \underline{t}_i \|^2 / d_{\max}^2}$,

$\underline{t}_i$ = the ith chosen center, and

$d_{\max}$ = the maximum distance between chosen centers.

The output weights are determined as follows:

Let

$\underline{w} = [w_1, w_2, \cdots, w_m]^T$,

$\underline{\psi}_k = [\psi_1(\underline{x}_k), \psi_2(\underline{x}_k), \cdots, \psi_m(\underline{x}_k)]^T$,

$R = E[\underline{\psi}_k \underline{\psi}_k^T]$, and $P = E[d_k \underline{\psi}_k]$.

Then,

$\underline{w}^* = R^{-1} P$

The distribution of $\underline{t}_i$ is important to the performance of RBFN.

From this point of view, this method does not give the stable performance.

Case 2: k-means clustering algorithm to select kernel centers (Moody and Darken, 1989)

Estimation function is represented by

$$\phi(\underline{x}) = \sum_{i=1}^{m} w_i \psi_i(\underline{x})$$

where

$\psi_i(\underline{x}) = e^{-\| \underline{x} - \underline{t}_i \|^2 / \sigma_i^2}$,

$\underline{t}_i$ = the ith chosen center, and

σ_i = the ith kernel width.

Phase 1: k-means clustering algorithm to select kernel centers

Step 1. Let $\{\underline{t}_k(n)\}_{k=1}^{m}$ denotes the kernel centers at the nth iteration. Choose $\underline{t}_k(0)$ randomly.

Step 2. Draw a sample vector $\underline{x}$ from a sample set S with a certain probability.

Step 3. Find the index of the best matching:

$$k(\underline{x}) = \arg\min_k \| \underline{x}(n) - \underline{t}_k(n) \| \quad \text{for } k = 1, \cdots, m.$$

Step 4. Update the kernel centers:

$$\underline{t}_k(n+1) = \begin{cases} \underline{t}_k(n) + \eta[\underline{x}(n) - \underline{t}_k(n)] & \text{if } k(\underline{x}) = k \\ \underline{t}_k(n) & otherwise \end{cases}$$

Step 5. If there is no noticeable change in $\underline{t}_k$s, stop. Otherwise, $n \leftarrow n+1$ and go to Step 2.

Phase 2: Determine kernel widths: for the ith kernel function

$$\psi_i(\underline{x}) = e^{-\| \underline{x} - \underline{t}_i \|^2 / \sigma_i^2},$$

choose the kernel width such as

$$\sigma_i \propto \min_{j \neq i} \| \underline{t}_i - \underline{t}_j \| \quad \text{for } j = 1, \cdots, m.$$

Phase 3: Determine the output weights:

$$\underline{w}^* = R^{-1} P$$

In this method, kernel centers are distributed according to the distribution of samples. However, it does not consider the output of the target function.

cf. Applying genetic algorithm (GA) for selecting the kernel centers (Billings, 1995):
GA is the global search algorithm and it tends to find the optimal distribution of kernel centers. However, this method requires exhaustive search of kernel centers.

Case 3: Supervised selection of kernel centers (Lee and Kil, 1991; Kil, 1993)

Estimation function is represented by

$$\phi(\underline{x}) = \sum_{i=1}^{m} w_i \psi_i(\underline{x})$$

where

$$\psi_i(\underline{x}) = e^{-\|\underline{x}-\underline{t}_i\|^2/\sigma_i^2},$$

$\underline{t}_i$ = the ith chosen center, and

σ_i = the ith kernel width.

Learning algorithm of RBFN

Phase 1: recruiting the necessary number of kernel functions

Step 1. Construct a RBFN with a small number of kernel functions (1 or 2), that is, $i\,(kernel\ index) = 1 \text{ or } 2$.

Step 2. For each sample, do the following procedure:

(1) if $\underline{x}_k$ generates big error (or maximum error for $k = 1, \cdots, n$), $i \leftarrow i+1$, $\underline{t}_i = \underline{x}_k$, and

$\sigma_i \propto \min_{j \neq i} \| \underline{t}_i - \underline{t}_j \|$ for $j = 1, \cdots, i$

(2) update output weights: weight parameters are update to incorporate the given sample $(\underline{x}_k, y_k)$.

(3) if satisfied, go to the next phase of learning. Otherwise, go to Step 2.

Phase 2: fine tuning of parameters of RBFN

Here, we assume that the weight vector $\underline{w}$ is near the optimal weight vector $\underline{w}^*$. Each output sample can be represented by

$$y_k = \hat{y_k}(\underline{w}) + n_k$$

where n_k represents Gaussian noise. An estimator of output $\hat{y_k}$ can be approximated as

$$\hat{y_k}(\underline{w}) \approx \hat{y_k}(\underline{w}_0) + [\frac{\partial \hat{y_k}}{\partial \underline{w}}|_{\underline{w}=\underline{w}_0}]^T (\underline{w} - \underline{w}_0).$$

Let

$$\begin{aligned}\tilde{y}_k &= y_k - (\hat{y}_k(\underline{w}_0) - [\frac{\partial \hat{y}_k}{\partial \underline{w}}|_{\underline{w}=\underline{w}_0}]^T \underline{w}_0) \\ &= [\frac{\partial \hat{y}_k}{\partial \underline{w}}|_{\underline{w}=\underline{w}_0}]^T \underline{w} + n_k \\ &= \underline{h}_k^T \underline{w} + n_k\end{aligned}$$

where

$$\underline{h}_k = \frac{\partial \hat{y}_k}{\partial \underline{w}}|_{\underline{w}=\underline{w}_0}.$$

Then, we can apply the RLS algorithm for the above equation.

The update rule for weight vector:

$$\begin{aligned}\underline{w}_{k+1} &= \underline{w}_k + \underline{a}_k(\tilde{y}_k - \underline{h}_k^T \underline{w}_k) \\ &= \underline{w}_k + \underline{a}_k(y_k - (\hat{y}_k(\underline{w}_0) + \underline{h}_k^T(\underline{w}_k - \underline{w}_0))) \\ &= \underline{w}_k + \underline{a}_k(y_k - \hat{y}_k(\underline{w}_k))\end{aligned}$$

$$\underline{a}_k = B_k \underline{h}_k$$

$$B_k = B_{k-1} - \frac{(B_{k-1}\underline{h}_k)(B_{k-1}\underline{h}_k)^T}{1 + \underline{h}_k^T B_{k-1} \underline{h}_k}$$

- Optimal structure of RBFNs

. Regularization methods: the complexity term is included in the error function and one of the supervised learning algorithms is applied. eg. weight decay or weight elimination method.
. Optimization methods: learning problem is reformulated as the optimization problem. eg. support vector machine (SVM) with kernel functions.
. Model selection methods: the confidence interval term indicating the bound on the difference between the true and empirical risks is derived and applied to the optimization of regression models. eg. AIC, BIC, SEB, MSMC, etc.

- Applications of RBFNs

. probability estimation
. pattern recognition
. function approximation
. nonlinear time series prediction
. dynamic reconstruction of chaotic processes