Radial-Basis Function Networks

v 1.0 – March 2002

Radial-Basis Function Networks

- Origin: Cover's theorem
- Interpolation problem
- Regularization theory
- Generalized RBFN
  - Universal approximation
  - Comparison with MLP
  - RBFN = kernel regression
- Learning
  - Centers
  - Widths
  - Multiplying factors
  - Other forms

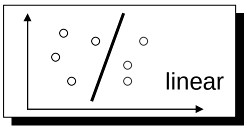
Origin: Covers' theorem

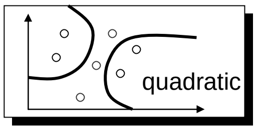
- Covers' theorem on separability of patterns (1965)
- $\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^P$ assigned to two classes C^1 C^2
- φ-separability:

$$\exists \boldsymbol{w} \mid \begin{cases} \boldsymbol{w}^T \varphi(\boldsymbol{x}) > 0 & \boldsymbol{x} \in C^1 \\ \boldsymbol{w}^T \varphi(\boldsymbol{x}) < 0 & \boldsymbol{x} \in C^2 \end{cases}$$

- Cover's theorem:
  - non-linear functions $\varphi(\boldsymbol{x})$
  - dimension hidden space > dimension input space

  $\rightarrow$ probability of separability closer to 1
- Example





Interpolation problem

- Given points $(\boldsymbol{x}^k, t^k)$, $\boldsymbol{x}^k \in \Re^d$, $t^k \in \Re$, $1 \le k \le P$:
- Find $F : \Re^d \rightarrow \Re$ that satisfies

$$F\left(\boldsymbol{x}^k\right) = t^k, \; k = 1 \ldots P$$

- RBF technique (Powell, 1988):

$$F(\boldsymbol{x}) = \sum_{k=1}^{P} w_k \varphi\left(\left\| \boldsymbol{x} - \boldsymbol{x}^k \right\|\right)$$

  - $\varphi\left(\left\| \boldsymbol{x} - \boldsymbol{x}^k \right\|\right)$ are arbitrary non-linear functions (RBF)
  - as many functions as data points
  - centers fixed at known points $\boldsymbol{x}^k$

Interpolation problem

$$F(\boldsymbol{x}^k) = t^k \qquad\qquad F(\boldsymbol{x}) = \sum_{k=1}^{P} w_k \varphi\left(\left\|\boldsymbol{x} - \boldsymbol{x}^k\right\|\right)$$

$$\begin{bmatrix} \varphi_{11} & \varphi_{12} & \cdots & \varphi_{1P} \\ \varphi_{21} & \varphi_{22} & \cdots & \varphi_{2P} \\ \vdots & \vdots & \ddots & \vdots \\ \varphi_{P1} & \varphi_{P2} & \cdots & \varphi_{PP} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_P \end{bmatrix} = \begin{bmatrix} t^1 \\ t^2 \\ \vdots \\ t^P \end{bmatrix}$$

where $\varphi_{kl} = \varphi\left(\left\|\boldsymbol{x}^k - \boldsymbol{x}^l\right\|\right)$

- Into matrix form: $\Phi \boldsymbol{w} = \boldsymbol{x} \longrightarrow \boldsymbol{w} = \Phi^{-1}\boldsymbol{x}$
- Vital question: is Φ non-singular ?

Michelli's theorem

- If points $\boldsymbol{x}^k$ are distinct, Φ is non-singular (regardless of the dimension of the input space)
- Valid for a large class of RBF functions:

$$\varphi(\boldsymbol{x}) = \sqrt{\left\|\boldsymbol{x} - \boldsymbol{c}\right\|^2 + k^2} \qquad (k > 0)$$

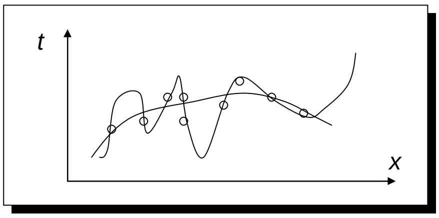
non-localized function

$$\varphi(\boldsymbol{x}) = \frac{1}{\sqrt{\left\|\boldsymbol{x} - \boldsymbol{c}\right\|^2 + k^2}}$$

$$\varphi(\boldsymbol{x}) = \exp\left(-\frac{\left\|\boldsymbol{x} - \boldsymbol{c}\right\|^2}{2\sigma^2}\right) \qquad (\sigma > 0)$$

localized functions

Learning: ill-posed problem



- Necessity for *regularization*
- Error criterion:

$$E(F) = \frac{1}{2P} \sum_{k=1}^{P} \left(t^k - F\left(\boldsymbol{x}^k\right) \right) + \lambda \frac{1}{2} C(\boldsymbol{w})$$

MSE regularization

Solution to the regularization problem

- Poggio & Girosi (1990):
  - if $C(w)$ is a (problem-dependent) linear differential operator, the solution to

$$E(F) = \frac{1}{2P} \sum_{k=1}^{P} \left(t^k - F\left(\boldsymbol{x}^k\right) \right) + \lambda \frac{1}{2} C(\boldsymbol{w})$$

is of the following form:

$$F(\boldsymbol{x}) = \sum_{k=1}^{P} w_k G\left(\boldsymbol{x}, \boldsymbol{x}^k\right)$$

where $G()$ is a Green's function,

$\boldsymbol{w} = (\boldsymbol{G} + \lambda \boldsymbol{I})^{-1} \boldsymbol{t}$

$G_{kl} = \boldsymbol{G}(\boldsymbol{x}^k, \boldsymbol{x}^l)$

Interpolation - Regularization

- Interpolation

$$F(\boldsymbol{x}) = \sum_{k=1}^{P} w_k \varphi\left(\left\| \boldsymbol{x} - \boldsymbol{x}^k \right\|\right)$$

$$\boldsymbol{w} = \Phi^{-1}\boldsymbol{x}$$

  - Exact interpolator
  - Possible RBF:

$$\varphi\left(\boldsymbol{x}, \boldsymbol{x}^k\right) = \exp\left(-\frac{\left\|\boldsymbol{x} - \boldsymbol{x}^k\right\|^2}{2\sigma^2}\right)$$

- Regularization

$$F(\boldsymbol{x}) = \sum_{k=1}^{P} w_k G\left(\boldsymbol{x}, \boldsymbol{x}^k\right)$$

$$\boldsymbol{w} = (\boldsymbol{G} + \lambda \boldsymbol{I})^{-1}\boldsymbol{t}$$

  - Exact interpolator
  - Equal to the « interpolation » solution iff λ=0
  - Example of Green's function:

$$G\left(\boldsymbol{x}, \boldsymbol{x}^k\right) = \exp\left(-\frac{\left\|\boldsymbol{x} - \boldsymbol{x}^k\right\|^2}{2\sigma^2}\right)$$

One RBF / Green's function for each learning pattern!

Generalized RBFN (GRBFN – RBFN)

- As many radial functions as learning patterns:
  - computationally (too) intensive (inversion of PxP matrix grows with P^3)
  - ill-conditioned matrix
  - regularization not easy (problem-specific)

→ *Generalized* RBFN approach!

$$F(\boldsymbol{x}) = \sum_{i=1}^{K} w_i \varphi\left(\left\| \boldsymbol{x} - \boldsymbol{c}_i \right\|\right)$$

Typically:

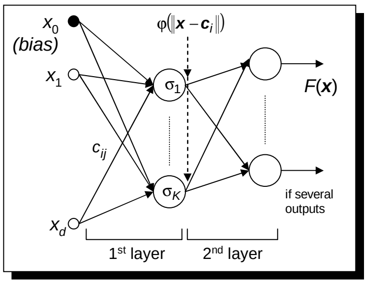
- $K << P$
- $\varphi\left(\left\|\boldsymbol{x} - \boldsymbol{c}_i\right\|\right) = \exp\left(-\frac{\left\|\boldsymbol{x} - \boldsymbol{c}_i\right\|^2}{2\sigma_i^2}\right)$

Parameters: $\boldsymbol{c}_i$, σ_i, w_i

Radial-Basis Function Networks (RBFN)

$$F(\boldsymbol{x}) = \sum_{i=1}^{K} w_i \varphi\left(\left\|\boldsymbol{x} - \boldsymbol{c}_i\right\|\right)$$

$$\varphi\left(\left\|\boldsymbol{x} - \boldsymbol{c}_i\right\|\right) = \exp\left(-\frac{\left\|\boldsymbol{x} - \boldsymbol{c}_i\right\|^2}{2\sigma_i^2}\right)$$



- Possibilities:
  - several outputs (common hidden layer)
  - bias (recommended) (see extensions)

RBFN: universal approximation

- Park & Sandberg 1991:
  - *For any continuous input-output mapping function f(**x**)*

$$\exists\ F(\boldsymbol{x}) = \sum_{i=1}^{K} w_i \varphi\left(\left\|\boldsymbol{x} - \boldsymbol{c}_i\right\|\right) \mid L_p\left(f(\boldsymbol{x}), F(\boldsymbol{x})\right) < \varepsilon \qquad \left(\varepsilon > 0, p \in [1, \infty]\right)$$

  - The theorem is stronger (radial summetry not needed)
  - K not specified
  - Provides a theoretical basis for *practical* RBFN!

RBFN and kernel regression

- non-linear regression model

  $t^k = f(\boldsymbol{x}^k) + \varepsilon^k = y^k + \varepsilon^k, \; 1 \le k \le P$
- estimation of $f(\boldsymbol{x})$: average of t around $\boldsymbol{x}$. More precisely:

$$f(\boldsymbol{x}) = E[y|\boldsymbol{x}] = \int_{-\infty}^{\infty} y f_Y(y|\boldsymbol{x}) dy = \frac{\int_{-\infty}^{\infty} y f_{\boldsymbol{X},Y}(\boldsymbol{x},y) dy}{f_{\boldsymbol{X}}(\boldsymbol{x})}$$

- Need for estimates of $f_{\boldsymbol{X},Y}(\boldsymbol{x},y)$ and $f_{\boldsymbol{X}}(\boldsymbol{x})$

  $\rightarrow$ Parzen-Rosenblatt density estimator

Parzen-Rosenblatt density estimator

$$\hat{f}_{\boldsymbol{X}}(\boldsymbol{x}) = \frac{1}{Ph^d} \sum_{k=1}^{P} K\left(\frac{\boldsymbol{x} - \boldsymbol{x}^k}{h}\right)$$

with $K()$ continuous, bounded, symmetric about the origin, with maximum value at 0, and with unit integral,
is consistent (asymptotically unbiased).

- Estimation of

$$\hat{f}_{\boldsymbol{X},Y}(\boldsymbol{x},y) = \frac{1}{Ph^{d+1}} \sum_{k=1}^{P} K\left(\frac{\boldsymbol{x} - \boldsymbol{x}^k}{h}\right) K\left(\frac{y - y^k}{h}\right)$$

RBFN and kernel regression

$$f(\mathbf{x}) = \frac{\int_{-\infty}^{\infty} y f_{\mathbf{X},Y}(\mathbf{x},y)dy}{f_{\mathbf{X}}(\mathbf{x})}$$

$$\hat{f}(\mathbf{x}) = \frac{\int_{-\infty}^{\infty} y \hat{f}_{\mathbf{X},Y}(\mathbf{x},y)dy}{\hat{f}_{\mathbf{X}}(\mathbf{x})} = \frac{\sum_{k=1}^{P} y^k K\left(\frac{\mathbf{x}-\mathbf{x}^k}{h}\right)}{\sum_{k=1}^{P} K\left(\frac{\mathbf{x}-\mathbf{x}^k}{h}\right)}$$

- Weighted average of y^i
- called Nadaraya-Watson estimator (1964)
- equivalent to *Normalized RBFN* in the unregularized context

RBFN ↔ MLP

- RBFN
  - single hidden layer
  - non-linear hidden layer
    linear output layer
  - argument of hidden units:
    Euclidean norm
  - universal approximation property
  - local approximators
  - splitted learning
- MLP
  - single or multiple hidden layers
  - non-linear hidden layer
    linear *or* non-linear output layer
  - argument of hidden units:
    scalar product
  - universal approximation property
  - global approximators
  - global learning

RBFN: learning strategies

$$F(\boldsymbol{x}) = \sum_{i=1}^{K} w_i \varphi(\|\boldsymbol{x} - \boldsymbol{c}_i\|) \qquad \varphi(\|\boldsymbol{x} - \boldsymbol{c}_i\|) = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{c}_i\|^2}{2\sigma_i^2}\right)$$

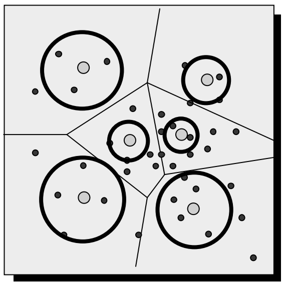
- Parameters to be determined: $\boldsymbol{c}_i$, σ_i, w_i
- Traditional learning strategy: splitted computation
  1. centers $\boldsymbol{c}_i$
  2. widths σ_i
  3. weights w_i

RBFN: computation of centers

- Idea: centers $\boldsymbol{c}_i$ must have the (density) properties of learning points $\boldsymbol{x}^k$
  $\rightarrow$ vector quantization
  - selected at random (in learning set)
  - competitive learning
  - frequency-sensitive learning
  - Kohonen maps
- This phase only uses the $\boldsymbol{x}^k$ information, not the t^k

RBFN: computation of widths

- Universal approximation property: valid with identical widths
- In practice (limited learning set): variable widths σ_i
- Idea: RBFN use *local* clusters



- choose σ_i according to standard deviation of clusters

RFBN: computation of weights

$$F(\boldsymbol{x}) = \sum_{i=1}^{K} w_i \varphi(\|\boldsymbol{x} - \boldsymbol{c}_i\|) \qquad \varphi(\|\boldsymbol{x} - \boldsymbol{c}_i\|) = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{c}_i\|^2}{2\sigma_i^2}\right)$$

constants !

- Problem becomes linear !
- Solution of least square criterion $E(F) = \frac{1}{2P}\sum_{k=1}^{P}\left(t^k - F\left(\boldsymbol{x}^k\right)\right)$ leads to

$$\boldsymbol{w} = \boldsymbol{\Phi}^{+}\boldsymbol{t} = \left(\boldsymbol{\Phi}^T\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^T$$

where

$$\boldsymbol{\Phi} \equiv \varphi_{ki} = \varphi\left(\left\|\boldsymbol{x}^k - \boldsymbol{c}_i\right\|\right)$$

- In practise: use SVD !

RBFN: gradient descent

$$F(\boldsymbol{x}) = \sum_{i=1}^{K} w_i \exp\left(-\frac{\left\|\boldsymbol{x}-\boldsymbol{c}_i\right\|^2}{2\sigma_i^2}\right)$$

1 **2** **3**

- 3-steps method: supervised unsupervised
- Once $\boldsymbol{c}_i$, σ_i, w_i have been set by the previous method, possibility of gradient descent on *all* parameters
- Some improvement, but
  - learning speed
  - local minima
  - risk of non-local basis functions
  - etc.

More elaborated models

- Add constant and linear terms

$$F(\boldsymbol{x}) = \sum_{i=1}^{K} w_i \exp\left(-\frac{\left\|\boldsymbol{x}-\boldsymbol{c}_i\right\|^2}{2\sigma_i^2}\right) + \sum_{i=1}^{d} w'_i \, x_i + w'_0$$

good idea (very difficult to approximate a constant with kernels…)

- Use normalized RBFN

$$F(\boldsymbol{x}) = \sum_{i=1}^{K} w_i \frac{\exp\left(-\frac{\left\|\boldsymbol{x}-\boldsymbol{c}_i\right\|^2}{2\sigma_i^2}\right)}{\sum_{j=1}^{K} \exp\left(-\frac{\left\|\boldsymbol{x}-\boldsymbol{c}_j\right\|^2}{2\sigma_j^2}\right)}$$

basis functions are bouded [0,1] $\rightarrow$ can be interpreted as probability values (classification)

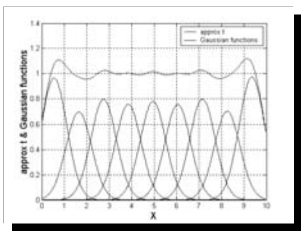
Back to the widths…

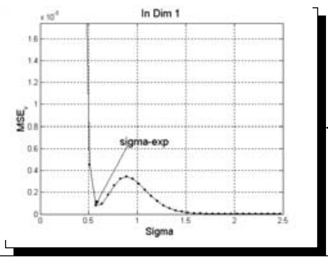
- choose σ_i according to standard deviation of clusters
- In the literature:
  - $\sigma = d_{\max} / \sqrt{2K}$ where $d_{\max}$ = maximum distance between centroids [1]
  - $\sigma_i = \frac{1}{p}\sqrt{\sum_{j=1}^{p} \left\| \boldsymbol{c}_i - \boldsymbol{c}_j \right\|^2}$ where index j scans the p nearest centroids to $\boldsymbol{c}_i$ [2]
  - $\sigma_i = r \min_j \left(\left\| \boldsymbol{c}_i - \boldsymbol{c}_j \right\| \right)$ where r is an overlap constant [3]
  - *.....*

[1] S. Haykin, *"Neural Networks a Comprehensive Foundation"*, Prentice-Hall Inc, second edition, 1999.
[2] J. Moody and C. J. Darken, *"Fast learning in networks of locally-tuned processing units"*, Neural Computation 1, pp. 281-294, 1989.
[3] A. Saha and J. D. Keeler, *"Algorithms for Better Representation and Faster Learning in Radial Basis Function Networks"*, Advances in Neural Information Processing Systems 2, Edited by David S. Touretzky, pp. 482-489, 1989.

Basic example

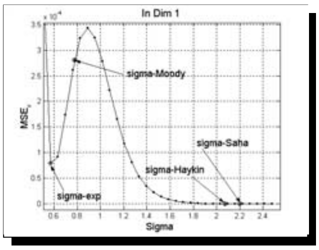
- Approximation of $f(\boldsymbol{x}) = 1$ with a d-dimensional RBFN
- In theory: identical w_i
- Experimentally: side effects
  $\rightarrow$ only middle taken into account

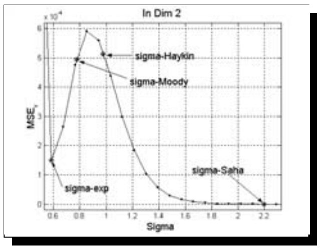


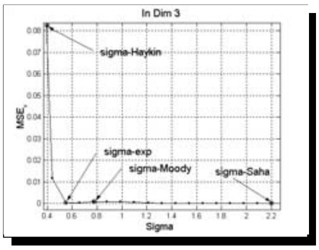
- 

Error versus width

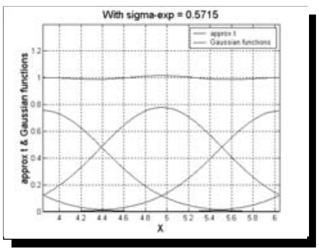
Basic example: erros vs space dimension

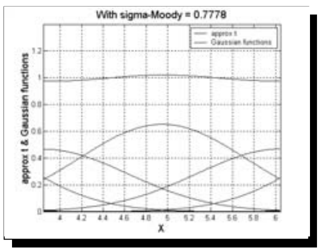


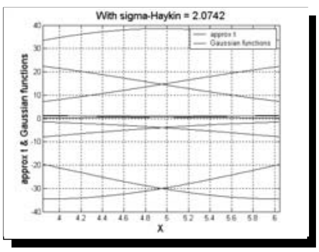




Basic example: local decomposition?

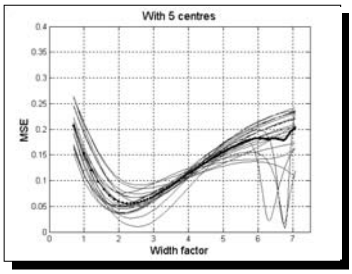






Multiple local minima in error curve

- Choose the first minimum to preserve the locality of clusters



- The first local minimum is usually less sensitive to variability

Some concluding comments

- RBFN: easy learning (compared to MLP)
  - in a cross-validation scheme: important!
- Many RBFN models
- Even more RBFN learning schemes…
- Results not very sensitive to unsupervised part of learning ($\boldsymbol{c}_i$, σ_i)
- Open work for a priori (proble-dependent) choice of widths σ_i

Sources and references

- Most of the basic concepts developed in these slides come from the excellent book:
  - Neural networks – a comprehensive foundation, S. Haykin, Macmillan College Publishing Company, 1994.
- Some supplementary comments come from the tutorial on RBF:
  - An overview of Radial Basis Function Networks, J. Ghosh & A. Nag, in: Radial Basis Function Networks 2, R.J. Howlett & L.C. Jain eds., Physica-Verlag, 2001.
- The results on the basic exemple were generated by my colleague N. Benoudjit, and are submitted for publication.