Faculty of Mathematics, Physics and Informatics
Comenius University in Bratislava

Neural Networks

Lecture 8

Radial-basis function networks

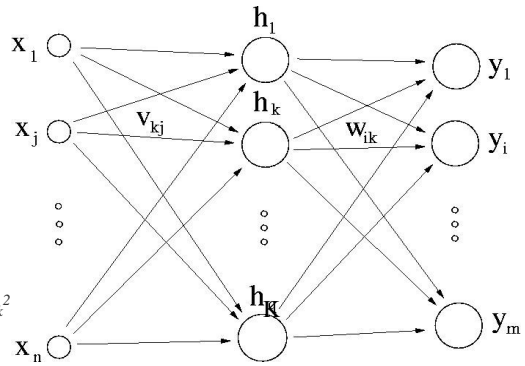
Combined NN models

- combination of unsupervised and supervised learning
- independent optimization, can be much faster than gradient descent, with similar results
- unsupervised learning → clustering
- more hidden units may be needed (compared to a completely supervised model)
- Examples:
  - counter-propagation networks (Hecht-Nielsen, 1987)
  - learning vector quantization (Kohonen, 1990)
  - radial-basis-function networks (Moody & Darken, 1989)

Radial-Basis-Function neural network

- Inputs $\boldsymbol{x}$, weights $\boldsymbol{w}$, outputs $\boldsymbol{y}$
- Output activation:

$$y_i = \sum_{k=1}^{K} w_{ik} h_k(\boldsymbol{x}) + w_{i0}$$



- h_k = radial activ. function, e.g.

  $h_k(\boldsymbol{x}) = \varphi_k(\|\boldsymbol{x}-\boldsymbol{v}_k\|) = \exp(-\|\boldsymbol{x}-\boldsymbol{v}_k\|^2/\sigma_k^2$

  $\boldsymbol{v}_k \sim$ center k, $\sigma_k \sim$ its width

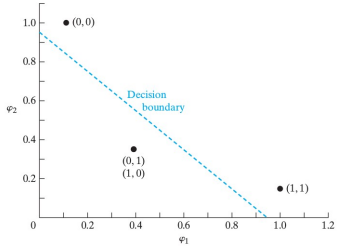
  $\varphi(d)$ are (usually) local functions because for $d \to \infty \;\; \varphi(d) \to 0$

  σ affects generalization

- $\boldsymbol{v}_k$ used for approximation of unconditional probability density of input data $p(\boldsymbol{x})$
- RBF as a receptive field (easier than that of an MLP)

Separability of patterns

- Data projection into high-dim. space:
  A complex pattern classification problem cast in a high-dim. space nonlinearly is more likely to be linearly separable than in a low-dim. space (*Cover, 1965*).
- Consider binary partitioning (dichotomy) for $\boldsymbol{x}_1,\boldsymbol{x}_2,...,\boldsymbol{x}_N$ (classes C_1,C_2). Dichotomy $\{C_1,C_2\}$ is $\boldsymbol{\phi}$-separable, where $\boldsymbol{\phi}(\boldsymbol{x}) = [\varphi_1(\boldsymbol{x}), \varphi_2(\boldsymbol{x}),...,\varphi_K(\boldsymbol{x})]$, if $\exists \boldsymbol{w} \in \Re^K$ such that for $\forall \boldsymbol{x} \in C_1$: $\boldsymbol{w}^T.\boldsymbol{\phi}(\boldsymbol{x}) > 0$ and for $\forall \boldsymbol{x} \in C_2$: $\boldsymbol{w}^T.\boldsymbol{\phi}(\boldsymbol{x}) < 0$.
- $\{\varphi_k(\boldsymbol{x})\}$ – feature functions (hidden space), $k = 1,2,...,K$
- Sometimes, non-linear transformation can result in linear separability without having to increase data dimension (e.g. XOR problem):



$\varphi_k(\boldsymbol{x}) = \exp(-\|\boldsymbol{x}-\boldsymbol{v}_k\|^2)$ $\quad \boldsymbol{v}_1 = [0\ 0]$, $\boldsymbol{v}_2 = [1\ 1]$

Input Pattern $\mathbf{x}$	First Hidden Function $\varphi_1(\mathbf{x})$	Second Hidden Function $\varphi_2(\mathbf{x})$
(1,1)	1	0.1353
(0,1)	0.3678	0.3678
(0,0)	0.1353	1
(1,0)	0.3678	0.3678

Interpolation problem

- Mapping data into higher dimensions can be useful:
- Then we can deal with multivariate interpolation in high-dim. space (Davis, 1963):

  Given the sets $\{\boldsymbol{h}_i \in \Re^K, d_i \in \Re\}$, find a function F that satisfies the condition: $F(\boldsymbol{h}_i) = d_i$, $i=1,2,...,N$. (in strict sense)
- For RBF, we get the set of linear equations: $\boldsymbol{w}^T \boldsymbol{h}_i = d_i$, $i = 1,2,...,N$.
- If $\mathbf{H}^{-1}$ exists, the solution is $\boldsymbol{w} = \mathbf{H}^{-1}\boldsymbol{d}$
- How can we be sure that interpolation matrix $\mathbf{H}$ is nonsingular?

- Theorem: Let $\{\boldsymbol{x}_i \in \Re^n\}$ be a set of distinct points $(i=1,2,...,N)$. Then $\mathbf{H}$ $[N \times N]$ with elements $h_{ij} = \varphi_{ij}(\|\boldsymbol{x}_i - \boldsymbol{x}_j\|)$, is nonsingular. (Michelli, 1986)
- a large class of RBFs satisfies this condition

Various types of radial-basis functions

Gaussian: $\varphi(r) = \exp(-r^2/\sigma^2)$

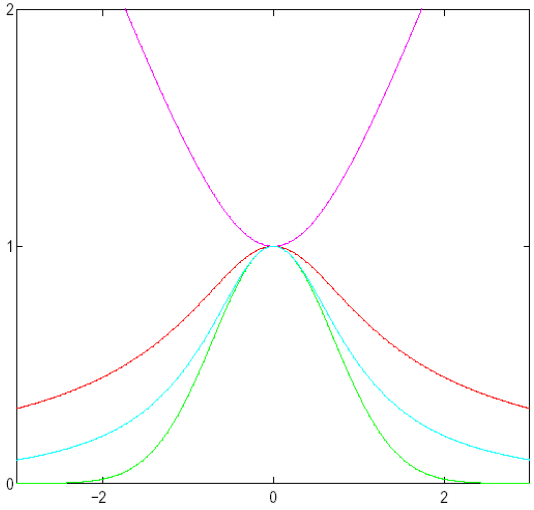
Multiquadrics: $\varphi(r) = (r^2+c^2)^{1/2}$

Inverse multiquadrics: $\varphi(r) = (r^2+c^2)^{-1/2}$

Cauchy: $\varphi(r) = 1/(1+r^2)$

$r \in \Re, \; c>0$

Matrix ***H*** for nonlocal multiquadrics is not positive definite. Despite that it can be used to approximate a smooth I/O mapping with greater accuracy than those that yield positive-definite interpolation matrix (Powell, 1988).



Training RBF networks

- two-stage process
- nonlinear (layer 1) and linear (layer 2) optimization strategies are applied to different learning tasks
- Approaches for layer 1:
  - fixed centers selected at random
  - self-organized selection of centers
- Approaches for layer 2
  - via pseudoinverse $\mathbf{H}^+$: then $\boldsymbol{w} = \mathbf{H}^+\boldsymbol{d}$
  - online stochastic optimization (delta rule),
  - online deterministic algorithm (RLS)
- Yet another method: supervised selection of centers and output weight setting (not described here)

Fixed centers selected at random

- "sensible" approach if training data are distributed in a representative manner:

  $G(\|\boldsymbol{x} - \boldsymbol{v}_j\|^2) = \exp(-K\|\boldsymbol{x} - \boldsymbol{v}_j\|^2/d^2_{\max})$

  K – number of centers, $d_{\max} = \max_{kl}\{\|\boldsymbol{v}_k - \boldsymbol{v}_l\|\}$, => $\sigma = d_{\max}/(2K)^{1/2}$
- RBFs become neither too flat nor too wide
- Alternative: individual widths σ_j, inversely proportional to density $p(\boldsymbol{x})$ – requires experimentation with data
- relatively insensitive to regularization, for larger data sets

Self-organized selection of centers

Self-organization: *K*-means clustering:

Initialization: randomize $\{\boldsymbol{v}_1(0), \boldsymbol{v}_2(0), ..., \boldsymbol{v}_K(0)\}$

Two steps: (until stopping criterion is met)

1. minimize $J(C) = min_{\{\boldsymbol{v}_k\}} \sum_{k=1}^{K} \sum_{C(i)=k} \|\boldsymbol{x}(i) - \boldsymbol{v}_k\|^2$ for given encoder C
   - by updating cluster centers: $\{\boldsymbol{v}_k(t)\}$
2. optimize the encoder: $C(i) = arg\, min_k \|\boldsymbol{x}(i) - \boldsymbol{v}_k\|^2$
   - by reassigning inputs to clusters

Given a set of N observations, find the encoder C that assigns these observations to the K clusters in such a way that, within each cluster, the average measure of dissimilarity of the assigned observations from the cluster mean is minimized.

- *no guarantee for finding an optimum*

Recursive Least Squares (RLS)

- RBF centers can be updated recursively
- How to compute optimal output weights, recursively, too?
- RLS algorithm summary: given $\{\boldsymbol{\phi}(p), d(p)\}$, $p = 1,2,...,N;\ p \equiv t$
- *Initialize:* $\boldsymbol{w}(0) = \mathbf{0}$, $\mathbf{P}(0) = \lambda^{-1}\mathbf{I}$, with $\lambda > 0$, $\lambda \approx 0$, regularizer $\frac{1}{2}\lambda\|\boldsymbol{w}\|^2$
- *Repeat:*

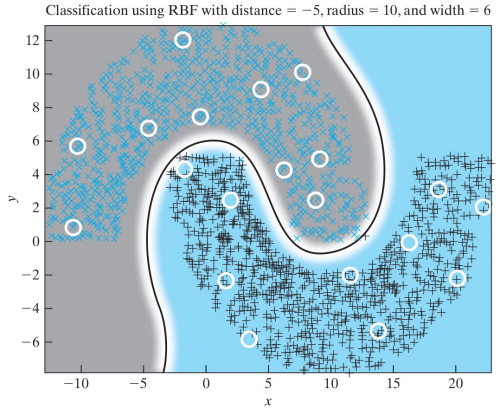
1. $\mathbf{P}(t) = \mathbf{P}(t-1) - \dfrac{\mathbf{P}(t-1)\boldsymbol{\Phi}(t)\boldsymbol{\Phi}^T(t)\mathbf{P}(t-1)}{1 + \boldsymbol{\Phi}^T(t)\mathbf{P}(t-1)\boldsymbol{\Phi}(t)}$
2. $\boldsymbol{g}(t) = \mathbf{P}(t).\boldsymbol{\phi}(t)$ (gain)
3. $a(t) = d(t) - \boldsymbol{w}^{\mathrm{T}}(t-1)\ \boldsymbol{\phi}(t)$ (prior estimation error)
4. $\boldsymbol{w}(t) = \boldsymbol{w}(t-1) + \boldsymbol{g}(t).a(t)$

Example using an RBF network

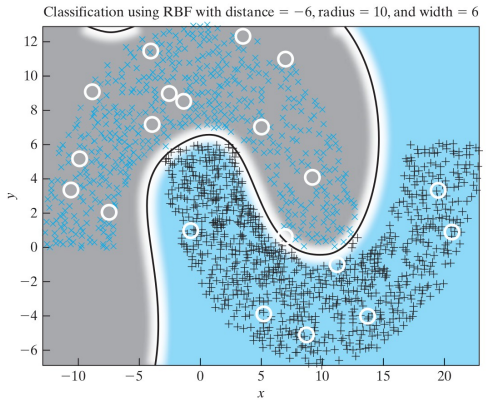
Two-moons classification task: 20 Gaussian units, 1000 points used for training, 2000 for testing. Different widths (σ) used.

$$\sigma = \frac{d_{\max}}{\sqrt{2K}}$$

$\sigma = 2.6$



$\sigma = 2.4$



Approximation properties of RBF networks

Theorem: (Park & Sandberg, 1991) Let $G: \Re^K \to \Re$ be an integrable bounded function such that G is continuous and $\int_{\Re^K} G(\boldsymbol{x})\, d\boldsymbol{x} \neq 0$. The family of RBF networks consists of functions $F: \Re^K \to \Re$:

$$F(\boldsymbol{x}) = \sum_{k=1}^{K} w_k\, G((\boldsymbol{x} - \boldsymbol{v}_k)/\sigma)$$

where $\sigma > 0$, $w_k \in \Re$ and $\boldsymbol{v}_k \in \Re^K$.

Then for any continuous function $f(\boldsymbol{x})$ there exists an RBF network with a set of centers $\boldsymbol{v}_k \in \Re^q$ and a common width $\sigma > 0$ such that $F(\boldsymbol{x})$ realized by RBF network is close to $f(\boldsymbol{x})$ in L_p norm, $p \in [1,\infty]$.

Note: Theorem does not require radial symmetry for kernel $G: \Re^K \to \Re$.

- Useful constraint in RBF design: $K < N$ (number of patterns)
- Gaussian centers as kernels: $\int_{\Re^K} G(\boldsymbol{x})\, d\boldsymbol{x} = 1$

Kernel $G(\boldsymbol{x})$ = continuous, bounded, and real function of $\boldsymbol{x}$, symmetric about the origin, where it attains its maximum value.

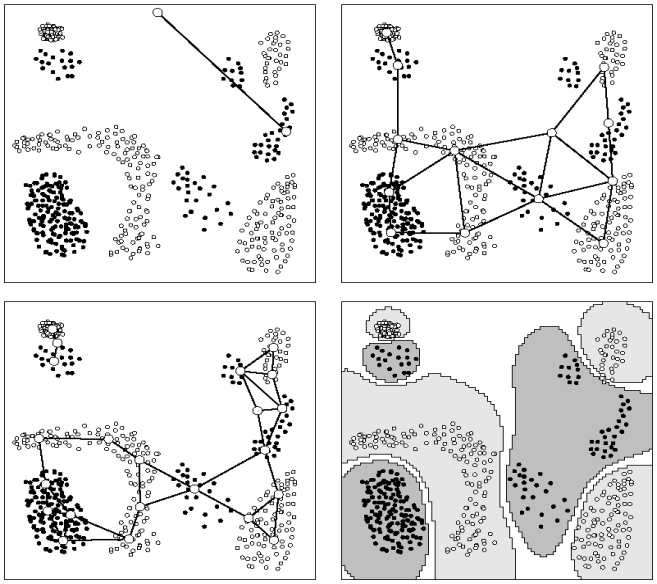
Comparison of RBF and MLP

- both are nonlinear layered feedforward networks
- both are universal approximators, using parametrized compositions of functions of single variables.
- localized vs. distributed representations on hidden layer =>
  - convergence of RBF may be faster
  - MLPs are global, RBF are local => MLP need fewer parameters
- different designs of a supervised network:
  - MLP = stochastic approximation problem
  - RBF = hypersurface-fitting problem in a high-dim. space
- one-stage (MLP) vs. two-stage (RBF) training scheme

Alternative self-organizing modules for center allocation

- Can be useful for input data
  - with varying dimensionality across input domain (e.g. Topology Representing Network)
  - with non-stationary distributions – dynamic networks (Dynamic Cell Structures, Growing CS)
- to be coupled with dynamic linear part
- all based on competitive learning

Example: binary classification with a growing RBF net



(Fritzke, 1994)

Summary

- RBF – hybrid feedforward NN model
  - first layer unsupervised, second layer supervised
- we deal with interpolation problem – as a hypersurface reconstruction problem
- various training algorithms for RBF centers
- RLS algorithm for computing output weights
- RBF as universal approximator
- applicable for function approximation and for classification