Neural Networks
Lecture Notes
Radial basis function networks

Marcel van Gerven

1 Learning Goals

After studying the lecture material you should:

1. understand the functional form of the Gaussian RBF network
2. understand how the RBF parameters are estimated

2 Notes

2.1 Comparison between MLPs and RBF networks

We have been using multilayer perceptrons (MLPs) as pattern classifiers. But in general, they are function approximators (depending on output layer nonlinearity and error function being minimized). As a function approximator, MLPs are nonlinear, semiparametric (can grow with number of hidden units) and universal. RBF networks are alternatives to MLPs for function approximation (we restrict ourselves to predicting continuous outputs).

In MLPs, the activation of hidden units is based on the dot product between the input vector and a weight vector:

$$f(\mathbf{x}^T\mathbf{w})\,.$$

In contrast, in RBF networks the activation of hidden units is based on the distance between the input vector and a prototype vector $\boldsymbol{\mu}$:

$$\phi_k(||\mathbf{x}-\boldsymbol{\mu}_k||)$$

where $||\mathbf{z}|| = (z_1^2 + \cdots + z_M^2)^{1/2}$. We will focus on Gaussian kernels as basis functions (others are possible):

$$\phi(a) = \exp\left(-\frac{a^2}{2\sigma^2}\right)\,.$$

Note that RBFs yield hidden units that depend on local parts of the input space. This is similar to the response properties of neurons. The nervous system contains many examples of neurons with "local" or "tuned" receptive fields, e.g.

- Orientation-selective cells in visual cortex.
- Somatosensory cells responsive to specific body regions.
- Cells in the barn owl auditory system tuned to specific inter-aural time delays.

There are several theoretical motivation for using RBFs:

- RBFs can be motivated using principles from function approximation, regularization theory, density estimation and interpolation in the presence of noise [Bis95]

- RBFs allow for a straightforward interpretation of the internal representation produced by the hidden layer

- Training algorithms for RBFs are significantly faster than those for MLPs

RBF networks are feed-forward networks consisting of a hidden layer of radial kernels and an output layer of linear neurons. The two RBF layers carry entirely different roles [Hay99]:

- The hidden layer performs a non-linear transformation of input space (typically of higher dimensionality than the input space)

- The output layer performs linear regression to predict the desired targets

One might ask why use a non-linear transformation followed by a linear one? Cover's theorem on the separability of patterns states:

> *A complex pattern-classification problem cast in a high-dimensional space non-linearly is more likely to be linearly separable than in a low-dimensional space.*

2.2 Training of RBF networks

We consider Gaussian RBF networks whose output is given by

$$y(\mathbf{x}) = \sum_{k=1}^{K} w_k \exp\left(-\frac{||\mathbf{x}-\boldsymbol{\mu}_k||^2}{2\sigma_k^2}\right)$$

If we want to use Gaussian RBFs to approximate a function specified by training data then the following questions arise:

1. How do we choose the Gaussian centers $\boldsymbol{\mu}_k$?

2. How do we determine the widths σ_k?

3. How do we determine the weights $\mathbf{w}$?

4. How do we select the number of basis functions K?

2.2.1 Determining cluster centers

In order to select the centers $\boldsymbol{\mu}_k$ a simple approach is to select K training data points at random as the centers. A potentially better way is to use unsupervised clustering, e.g. using K-means. This proceeds as follows:

- Iterate until convergence:
  - Assignment step:

    $$\mathcal{C}_k^{(t)} = \{\mathbf{x}\colon ||\mathbf{x}-\boldsymbol{\mu}_k^{(t)}||^2 \le ||\mathbf{x}-\boldsymbol{\mu}_j^{(t)}||^2 \;\; \forall j, 1 \le j \le K\}$$

    such that C_k contains the indices of the training data points closest to $\boldsymbol{\mu}_k$.
  - Update step:

    $$\boldsymbol{\mu}_k^{(t+1)} = \frac{1}{|\mathcal{C}_k^{(t)}|} \sum_{\mathbf{x}\in\mathcal{C}_k^{(t)}} \mathbf{x}$$

    where $|\cdot|$ denotes set size.

2.2.2 Determining cluster widths

Once cluster centers are determined, the variance for each basis function can be set to:

$$\sigma_k^2 = \frac{1}{|C_k|} \sum_{\mathbf{x} \in C_k} ||\mathbf{x} - \boldsymbol{\mu}_k||^2 \,.$$

A common simplification is to assume the same width for all basis functions:

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

where $d_{\max}$ is the maximal distance between centers and K is nr of clusters.

2.2.3 Determining the weights

With the hidden layer decided, weight training can be treated as a linear regression problem:

$$\boldsymbol{\Phi}\mathbf{w} = \mathbf{t}$$

where ϕ_{nk} is the output of the k-th basis function for the n-th training example and t_n is the target value for the n-th training example. This can be solved in one shot using the pseudo-inverse:

$$\mathbf{w} = \boldsymbol{\Phi}^\dagger \mathbf{t} = (\boldsymbol{\Phi}^T \boldsymbol{\Phi})^{-1} \boldsymbol{\Phi}^T \mathbf{t}$$

Note that a bias term needs to be included in $\boldsymbol{\Phi}$ (basis function with constant output). This procedure is fast since it does not need gradient descent (analytical solution).

2.2.4 Considerations

RBF training procedures have a major disadvantage since selection of RBF centers is not guided by the error function at the output layer. I.e. RBF centers that are representative of the feature space density are not guaranteed to capture the structure that carries discriminative information. To avoid this problem, fully-supervised algorithms can also be used for RBF training. We could use backpropagation to compute the required gradients $\frac{\partial E}{\partial \mu_k}$, $\frac{\partial E}{\partial \sigma_k}$, $\frac{\partial E}{\partial w_k}$. An advantage of this approach is that units are optimally tuned for producing correct outputs from the network. Issues are that it is much slower and the cluster widths can grow large, which implies that units are no longer locally tuned.

3 Reading material

For an overview on radial basis function networks, consult [Bis95].

References

[Bis95] C M Bishop. *Neural Networks for Pattern Recognition*. 1995.

[Hay99] S S Haykin. *Neural Networks: A Comprehensive Foundation*. International edition. Prentice Hall, 1999.

[KKSO84] Teuvo Kohonen, M Kai, Tapio Saram, and Others. Phonotopic maps-insightful representation of phonological features for speech recognition. 1984.