Charu C. Aggarwal
IBM T J Watson Research Center
Yorktown Heights, NY

Radial Basis Function Networks

Neural Networks and Deep Learning, Springer, 2018
Chapter 5

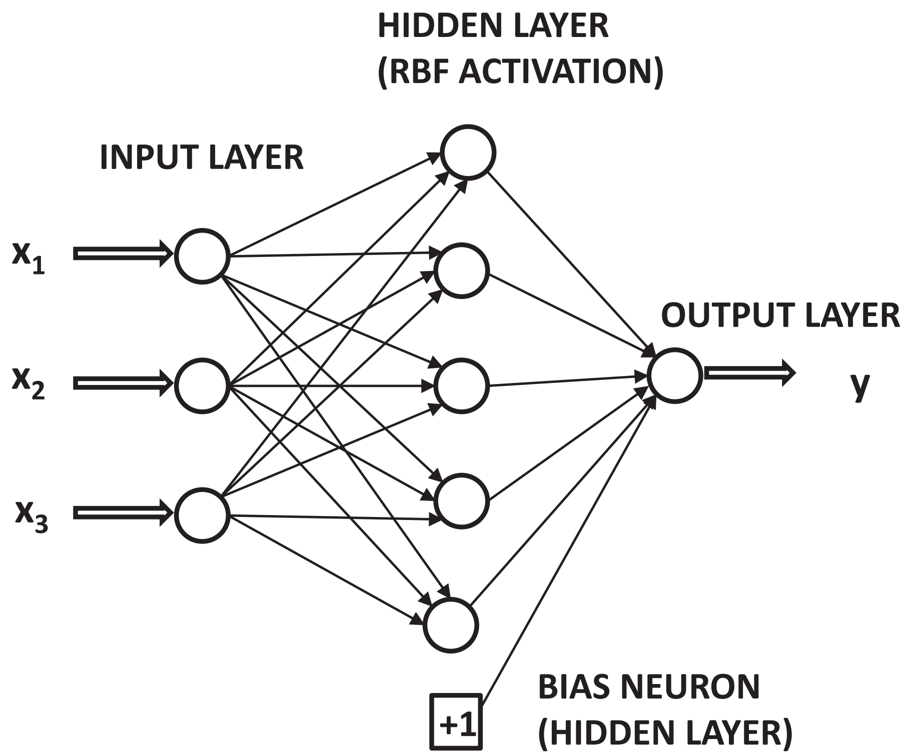
Radial Basis Function Networks

- Radial basis function (RBF) networks represent a fundamentally different paradigm in neural networks.
  - Not deep learners $\Rightarrow$ Often a single *unsupervised* hidden layer is used.
  - Deep learners represent an exercise in supervised feature engineering.
- RBF networks are closely related to SVMs.
  - SVMs represent a special case of RBF networks.
  - Like SVMs, RBF networks are universal function approximators.

When to Use RBF Networks

- Deep networks work best when the data has rich structure (e.g., images).
  - Property of hierarchical and supervised feature engineering.
- RBF networks are best when the data is noisy (but structure is less intricate).
  - Unsupervised feature engineering is robust to noise.

RBF Network



- Single (unsupervised) hidden layer with high dimensionality $m \gg d$ and linear output layer.

- Each hidden unit contains a prototype vector and activation depends on similarity of input to prototype (kernel similarity!).

Workings of the RBF Network

- Each of m hidden units has its own prototype vector $\overline{\mu}_i$ and bandwidth σ_i.
  - Common to set each $\sigma_i = \sigma$.

- For input vector $\overline{X}$, activation h_i of ith hidden unit (no weights!):

$$h_i = \Phi_i(\overline{X}) = \exp\left(-\frac{||\overline{X} - \overline{\mu}_i||^2}{2 \cdot \sigma_i^2}\right) \quad \forall i \in \{1, \ldots, m\} \qquad (1)$$

- Output layer is linear classifier/regressor with weights w_i.

$$\widehat{y} = \sum_{i=1}^{m} w_i h_i \text{ [Real-valued outputs]}$$

How do RBF Networks Classify Nonlinearly Separable Classes?

- Work on Cover's principle of separability of patterns.

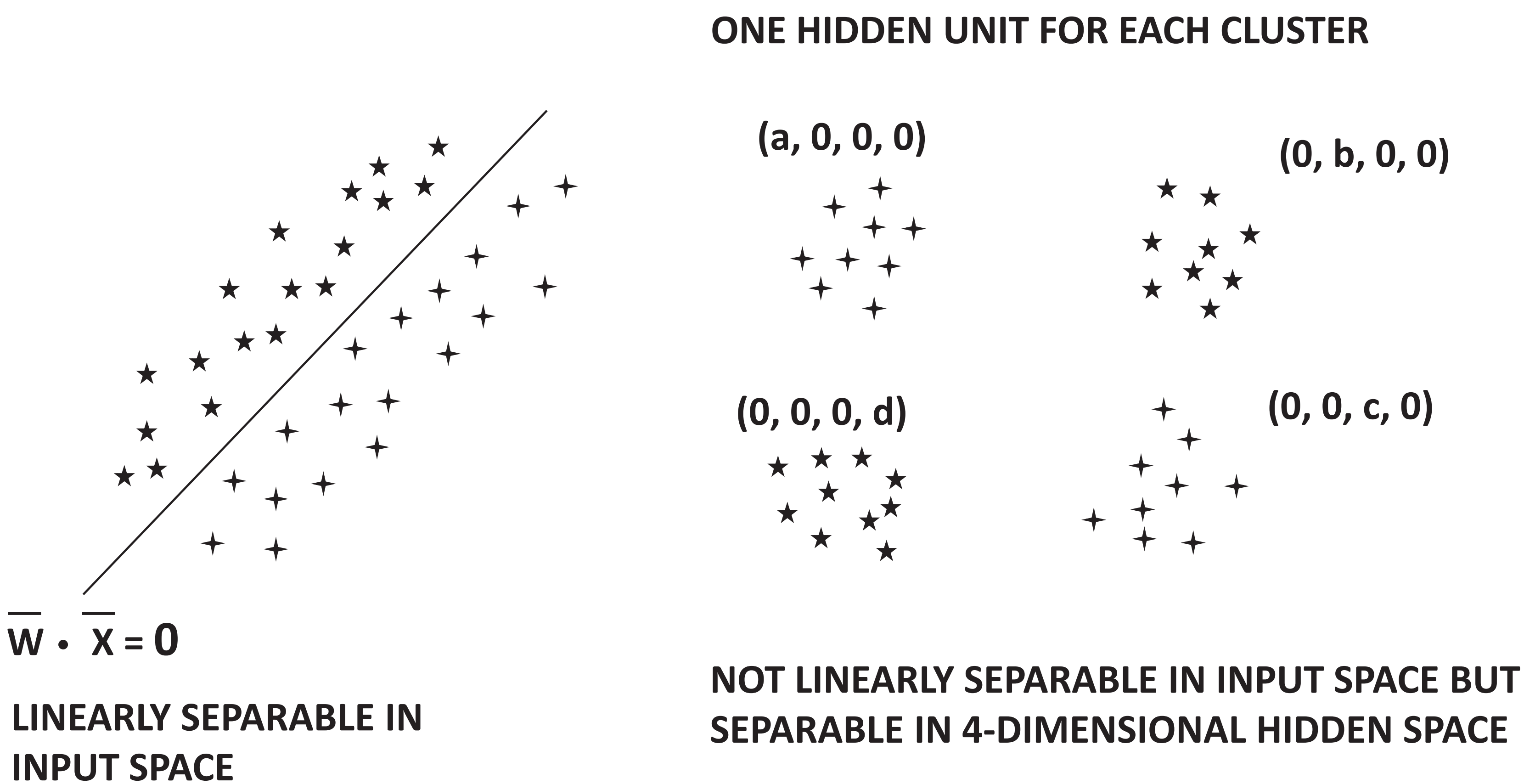
- Transforming low-dimensional data to high-dimensional space leads to greater ease in linear separation.

- The prototypes define local influence regions of the space.

  – Each feature corresponds to a local region.

- The final layer puts each region on the appropriate side of the separator.

Illustration of Separation Process



- One prototype from each cluster.
- Each local region is mapped to its own feature with a possible linear separator as $\overline{W} = [1, -1, 1, -1]$.

Training an RBF Network

- Training works in two phases:
  - Learn the prototype vectors $\overline{\mu}_i$ and bandwidth σ in an unsupervised manner.
  - Learn the weights of the output layer in supervised manner.
    - Straightforward training of single-layer network with engineered features.

Training the Hidden Layer

- Only need to find the prototype vectors $\overline{\mu}_i$ and bandwidth σ.
  - The prototypes can be sampled from data or can be centroids of clusters.

- Let d_{max} be maximum distance between pairs of prototypes and d_{ave} be average distance.
  - Two heuristic choices of σ are $d_{max}/\sqrt{m}$ and $2 \cdot d_{ave}$.
  - The bandwidth can also also be tuned on validation data.

Kernel Methods are Special Cases of RBF Networks

- Set the prototypes to all data points and:
  - Linear output layer (squared loss) for kernel regression/Fisher discriminant.
  - Linear output layer (hinge loss) for SVM
  - Logistic output layer (log loss) for kernel logistic regression
- Proofs in book.

Are Supervised Methods Any Good?

- Supervised training methods for hidden layer discussed in book.

- Generally, supervision of hidden layer leads to overfitting.

  - Supervised feature engineering is generally done by deep networks.

  - RBF networks are too shallow!

  - RBF prototype/bandwidth parameters have too complicated a loss surface to be learned in a supervised manner.

- Only mild forms of supervision desirable (e.g., tuning σ or mildly supervised prototype collection).