Statistical Methods in Artificial Intelligence
CSE471 - Monsoon 2016 : Lecture 08

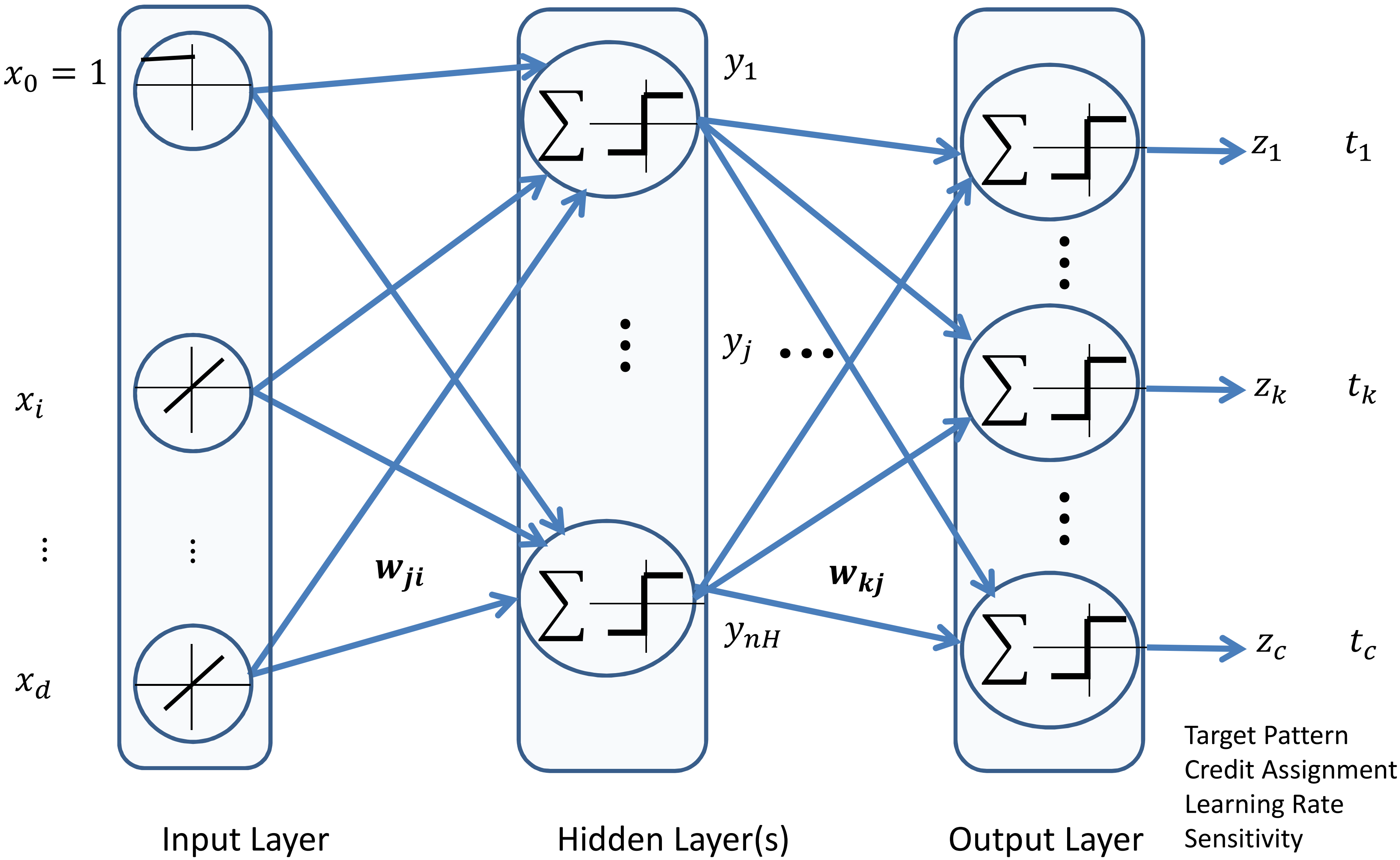
Avinash Sharma

CVIT, IIIT Hyderabad

Lecture Plan

- Recap
- Backpropagation as Feature Mapping
- Practical Aspects of Backpropagation
- Additional Networks
  - Deep Learning & Convolution Networks (ConvNet)
  - Recurrent Networks
  - RBF Networks

Backpropagation in NN

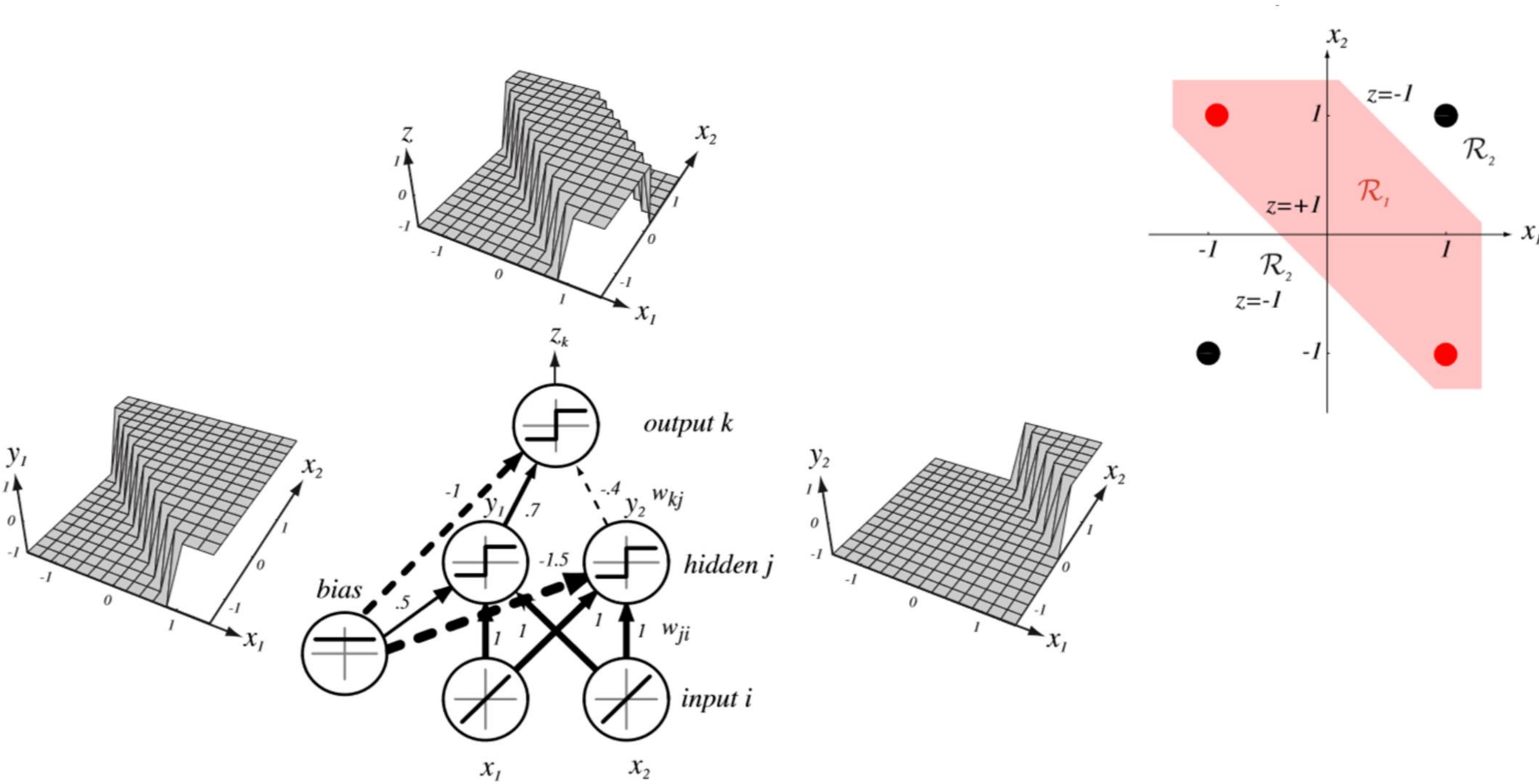


Backpropagation in Neural Networks

- $J(\mathbf{w}) = \frac{1}{2}\sum_{k=1}^{c}(t_k - z_k)^2 = \frac{1}{2}\|\mathbf{t} - \mathbf{z}\|^2$

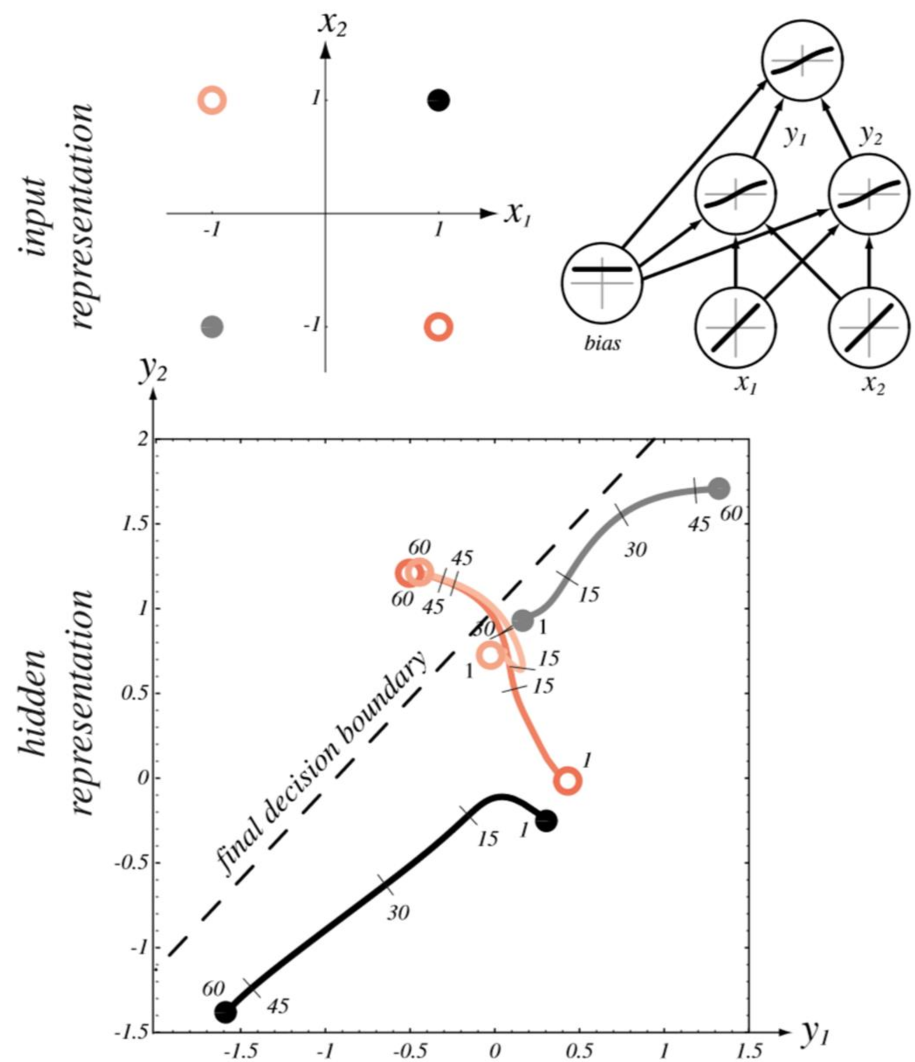
- $\Delta\mathbf{w} = -\eta\frac{\partial J}{\partial \mathbf{w}}, \Delta w_{pq} = -\eta\frac{\partial J}{\partial w_{pq}}$
- $\Delta w_{kj} = \eta\delta_k y_j = \eta(t_k - z_k)f'(net_k)y_j$
- $\Delta w_{ji} = \eta\delta_j x_i = \eta\left[\sum_{k=1}^{c} w_{kj}\,\delta_k\right]f'(net_j)x_i$

Modelling the Non-linearity



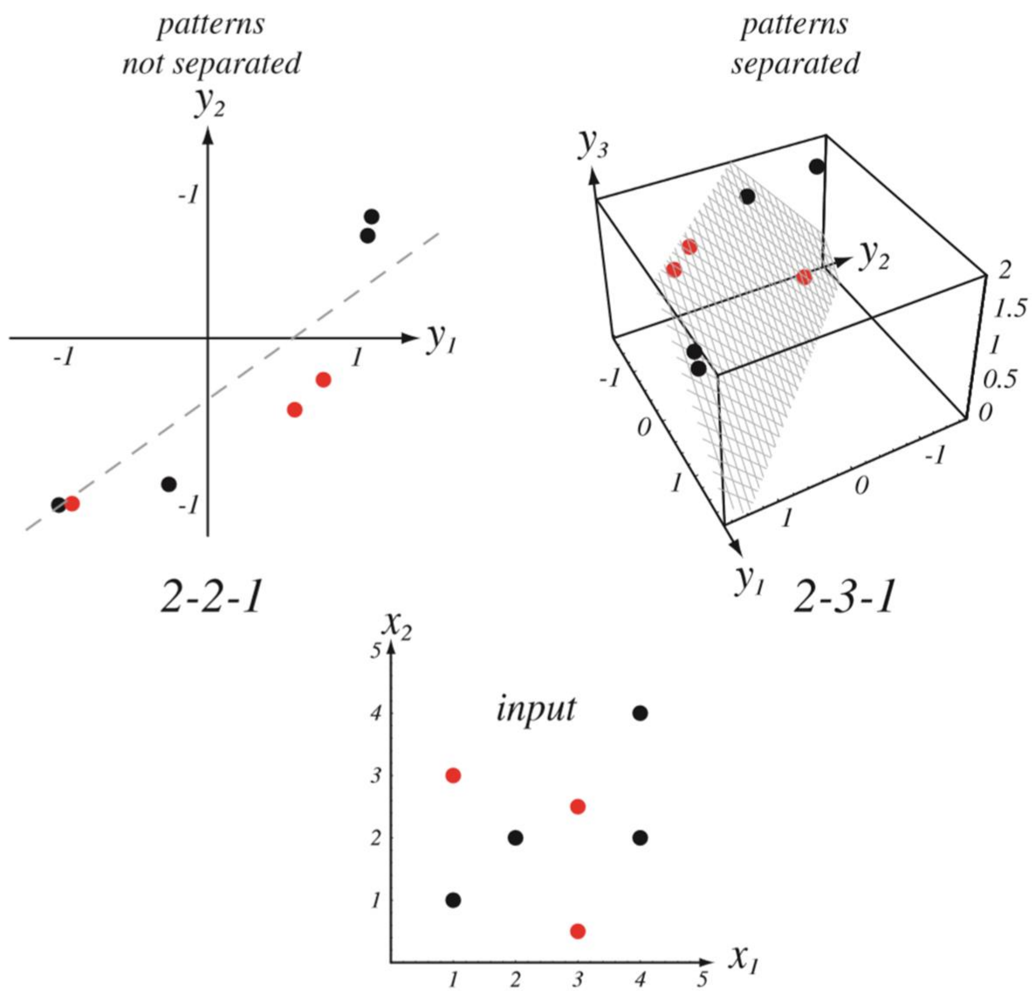
Backpropagation as Feature Mapping

- Output of hidden layers turns out to be linearly separable.
- Input-hidden layer achieves non-linear transform.
- Hidden-output layer feed forward only achieves a linear classification.



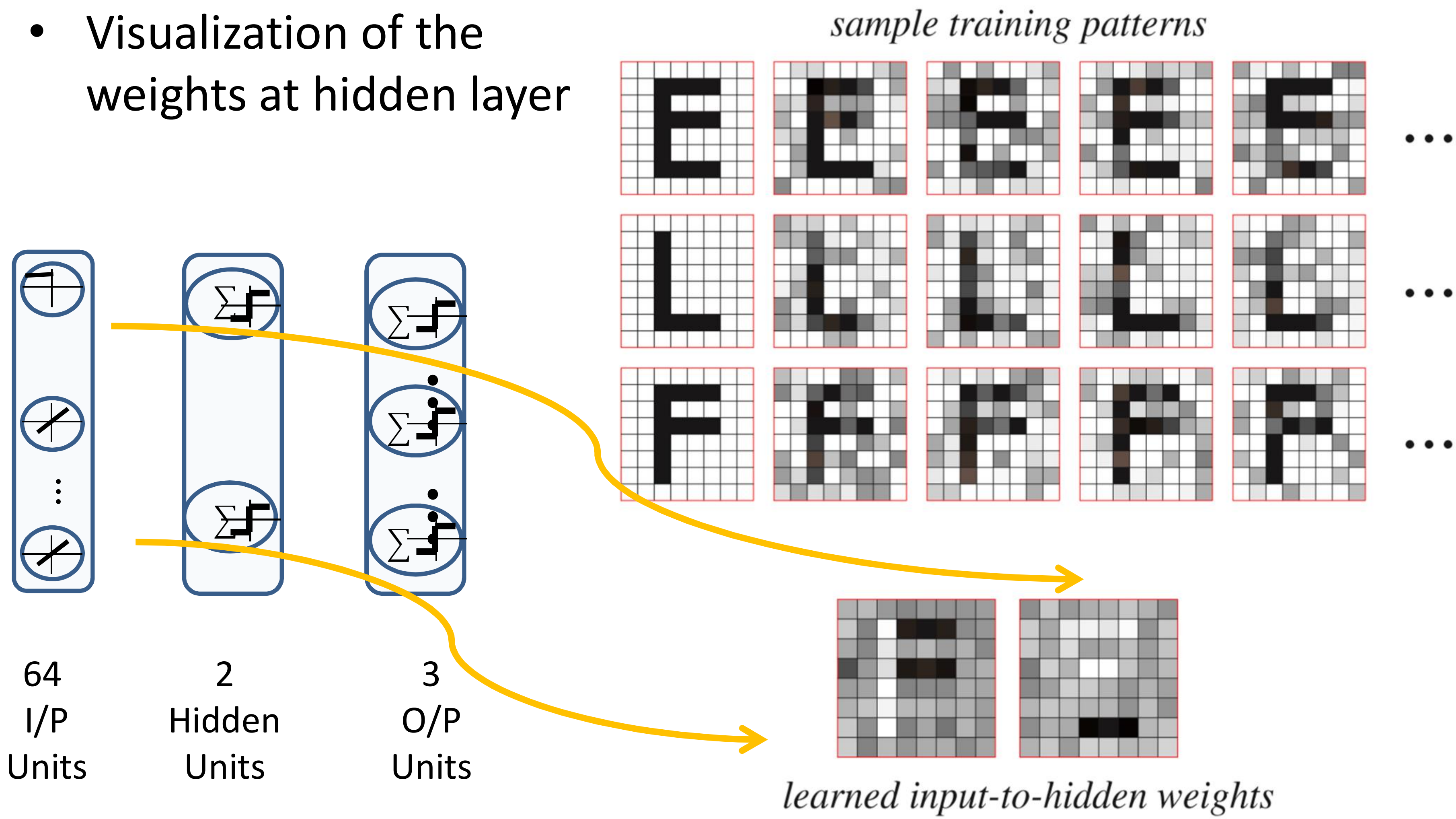
Backpropagation as Feature Mapping

- Output of hidden layers turns out to be linearly separable.
- Input-hidden layer achieves non-linear transform.
- Hidden-output layer feed forward only achieves a linear classification.
- Therefore, adding more hidden units might improve the performance



Backpropagation as Feature Mapping

- Visualization of the weights at hidden layer

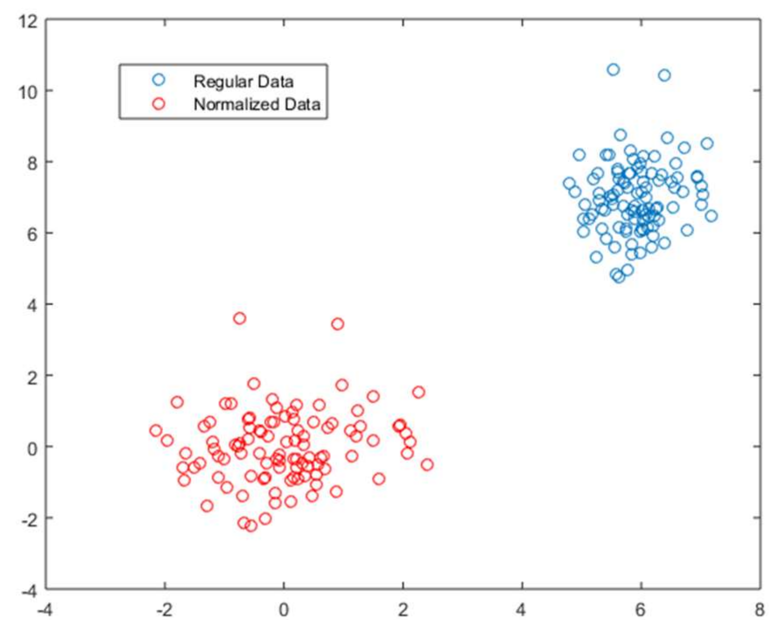


Practical Aspects of Backpropagation

- Activation Function
  - $f(\cdot)$ should be ***Non-linear***
  - $f(\cdot)$ should ***Saturate***
  - $f(\cdot)$ should be ***Continuous & Smooth***
  - $f'(\cdot)$ should be ***Defined***
  - $f(\cdot)$ can have ***Monotonicity***
  - $f(\cdot)$ can be ***linear for small values of*** $\boldsymbol{net}$

Practical Aspects of Backpropagation

- **Scaling of Input**
  - $\boldsymbol{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_m]^T{}_{m \times d}$
  - $\widetilde{\boldsymbol{X}} = \boldsymbol{X} - mean(\boldsymbol{X})$ (Centering of Data)
  - $\boldsymbol{X}_{Norm} = \widetilde{\boldsymbol{X}}/\boldsymbol{\sigma}$ (Column-wise division by Standard Deviation of each dimension)



Practical Aspects of Backpropagation

- **Scaling of Input**
  - $\boldsymbol{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_m]^T{}_{m \times d}$
  - $\widetilde{\boldsymbol{X}} = \boldsymbol{X} - mean(\boldsymbol{X})$ (Centering of Data)
  - $\boldsymbol{X}_{Norm} = \widetilde{\boldsymbol{X}}/\boldsymbol{\sigma}$ (Column-wise division by Standard Deviation of each dimension)
- **Target Values**
  - Use $+1$ and -1 or any real value in this range as output
  - Related to saturation value of the activation function
- **Training with Noise**
  - Add random noise to original training samples for generating more training samples

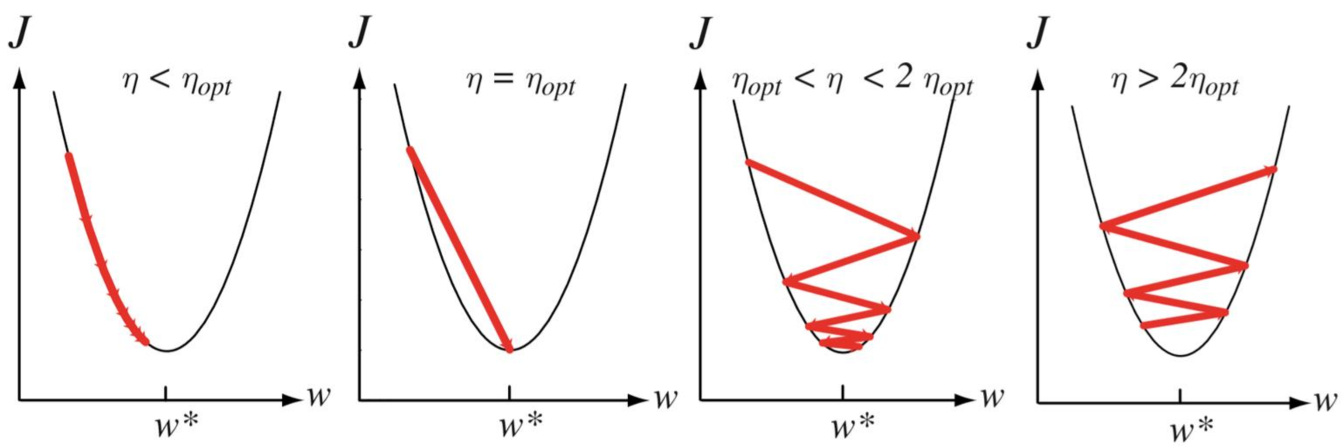
Practical Aspects of Backpropagation

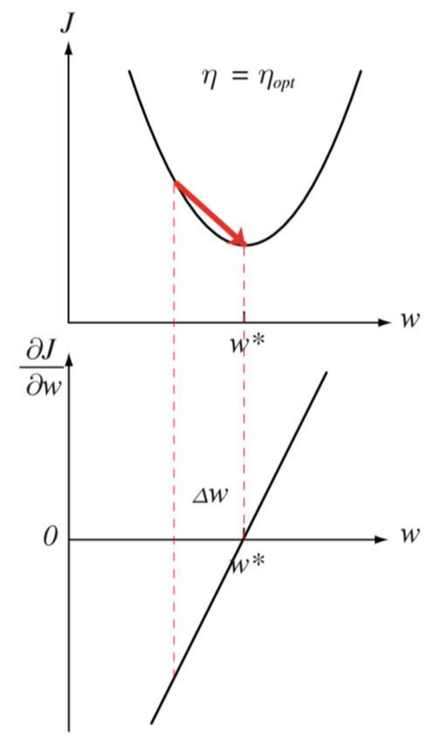
- **Manufacturing Data**
  - Add translation and rotation transforms to original training data to generate more rich training data samples
- **Number of Hidden Units**
  - Too few leads to high test error due to lack of expressibility
  - Too many leads to overfitting to training data
  - Choose such that total number of weights = $m/10$.
- **Weight Initialization**
  - Do not initialize with zero weights
  - Use both random positive & negative weights as data is standardized
  - $-1/\sqrt{d} < w_{ji} < +1/\sqrt{d}$ and $-1/\sqrt{nH} < w_{kj} < +1/\sqrt{nH}$

Practical Aspects of Backpropagation

- **Learning Rates**
  - For quadratic error criterion function J :

$$\eta_{opt} = \left(\frac{\partial^2 J}{\partial \mathbf{w}^2}\right)^{-1}$$





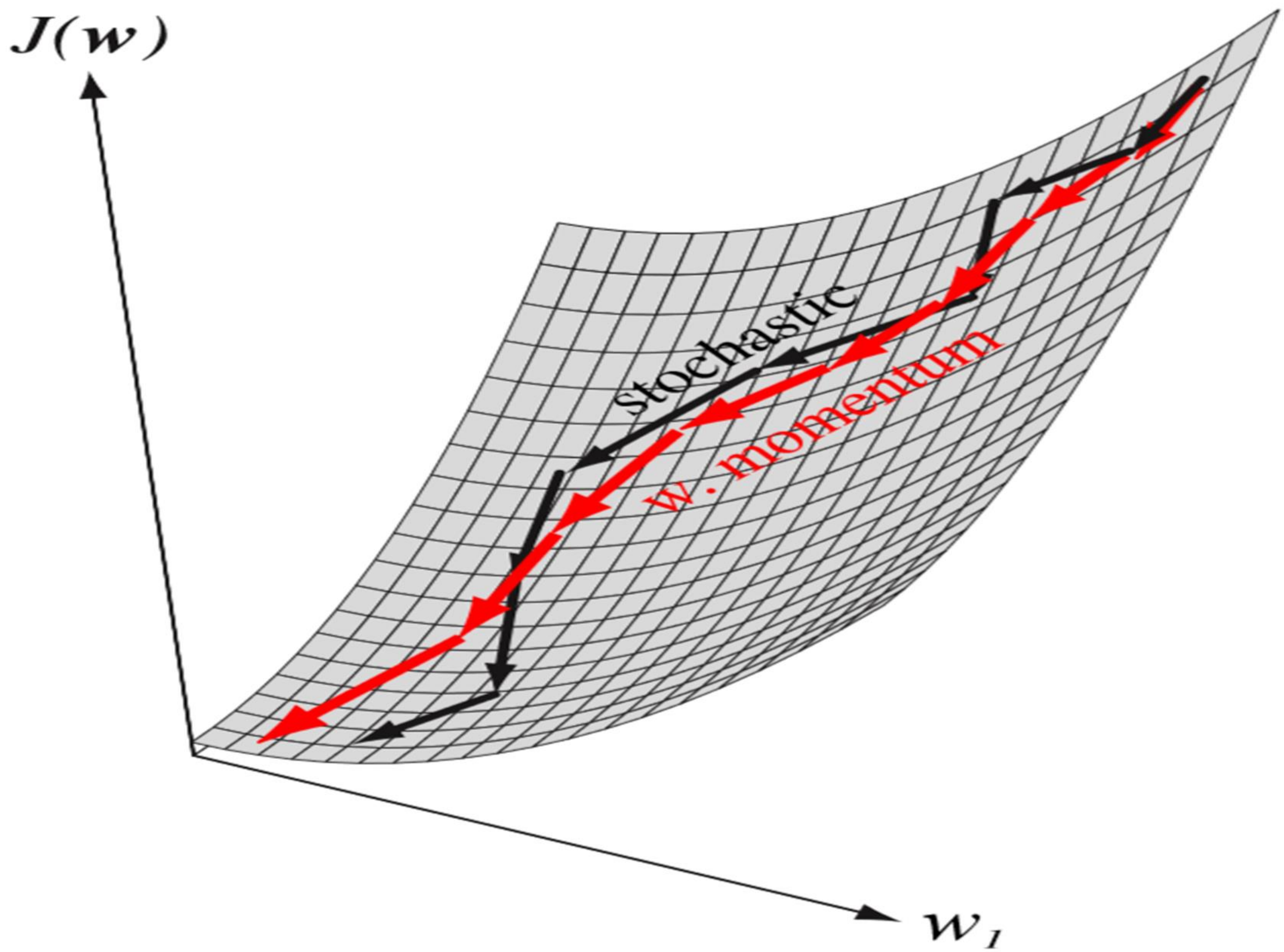
Practical Aspects of Backpropagation

- **Learning Rates**
  - For quadratic error criterion function J :

    $$\eta_{opt} = \left(\frac{\partial^2 J}{\partial \mathbf{w}^2}\right)^{-1}$$

- **Momentum**
  - Continue with inertia in each weight update
  - $\mathbf{w}(m+1) = \mathbf{w}(m) + (1-\alpha)\Delta\mathbf{w}(m) + \alpha\Delta\mathbf{w}(m-1)$

Practical Aspects of Backpropagation



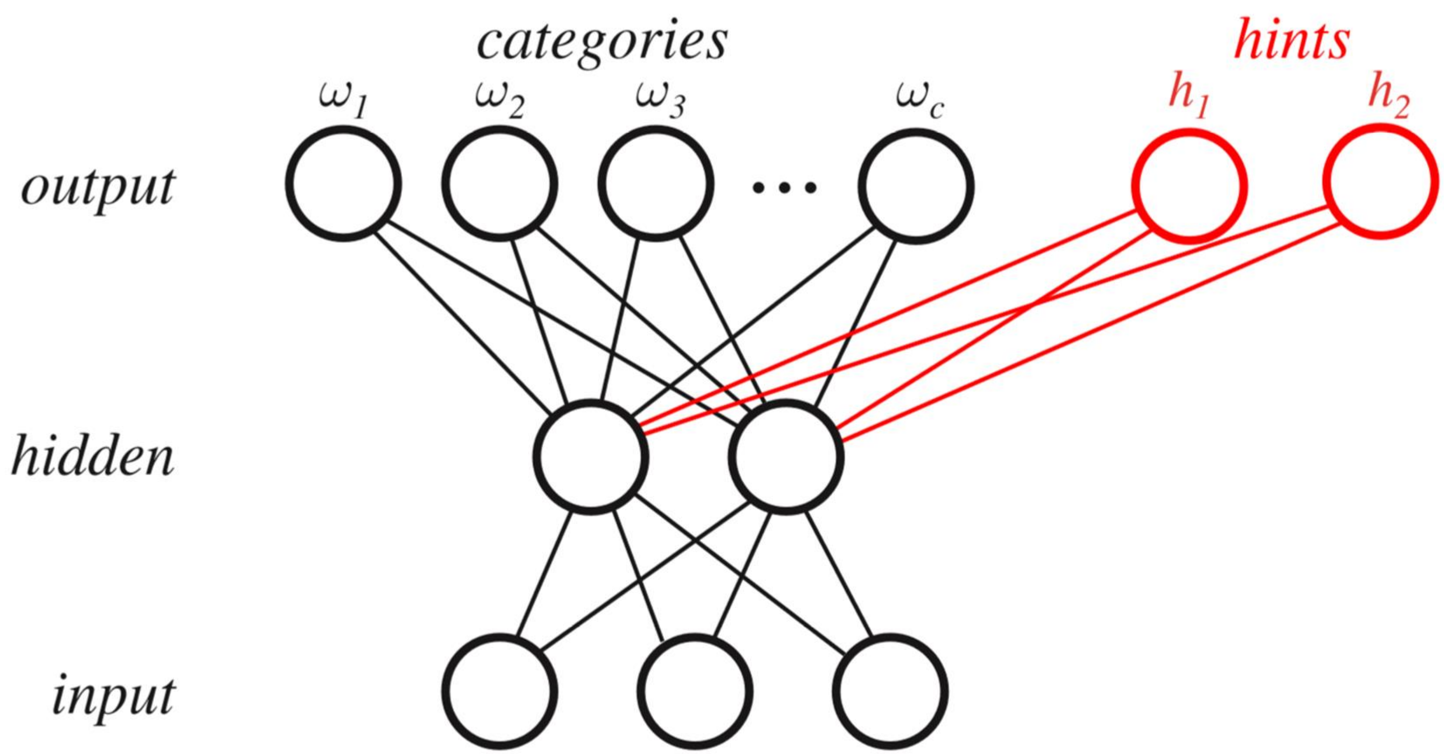
Practical Aspects of Backpropagation

- **Learning Rates**
  - For quadratic error criterion function J :

    $\eta_{opt} = \left(\frac{\partial^2 J}{\partial \mathbf{w}^2}\right)^{-1}$
- **Momentum**
  - Continue with inertia in each weight update
  - $\mathbf{w}(m+1) = \mathbf{w}(m) + (1-\alpha)\Delta\mathbf{w}(m) + \alpha\Delta\mathbf{w}(m-1)$
- **Weight Decay**
  - $w_{new} = w_{old}(1-\epsilon)$
  - Weights that do not affect the error function will eventually become zero.

Practical Aspects of Backpropagation

- Hints
  - Add ancillary units to output only for training phase.
  - In testing phase remove these extra units and related weights.



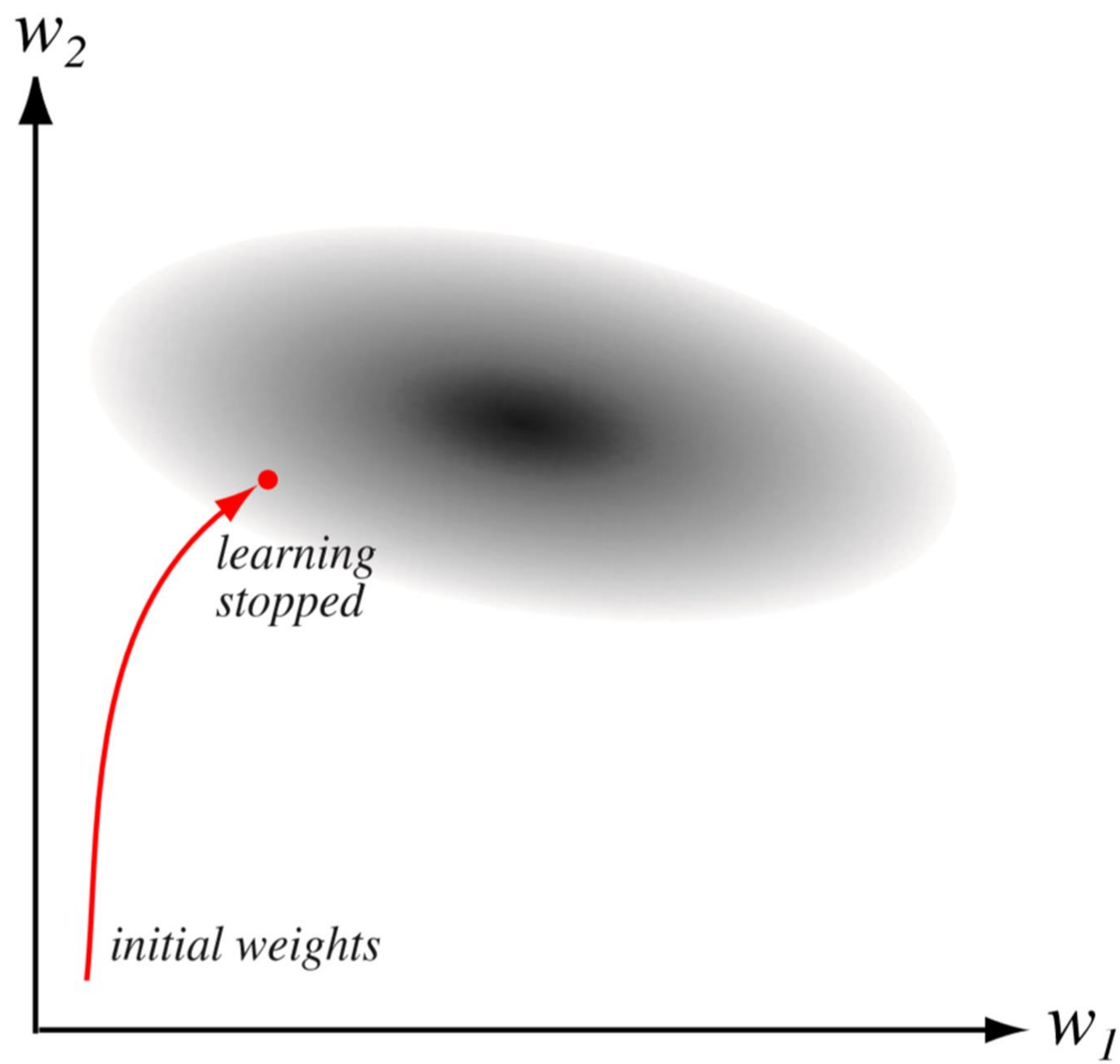
Practical Aspects of Backpropagation

- Hints
  - Add ancillary units to output only for training phase.
  - In testing phase remove these extra units and related weights
- Online/Batch/Stochastic Training
  - Stochastic training is mostly preferred over batch
  - Online is rarely used for tasks where storing all data samples if prohibitive

Practical Aspects of Backpropagation

- Hints
  - Add ancillary units to output only for training phase.
  - In testing phase remove these extra units and related weights
- Online/Batch/Stochastic Training
  - Stochastic training is mostly preferred over batch
  - Online is rarely used for tasks where storing all data samles if prohibitive
- Stopped Training
  - Excessive training can lead to poor generalization
  - Stopping training before low error reached is good to avoid overfitting

Practical Aspects of Backpropagation



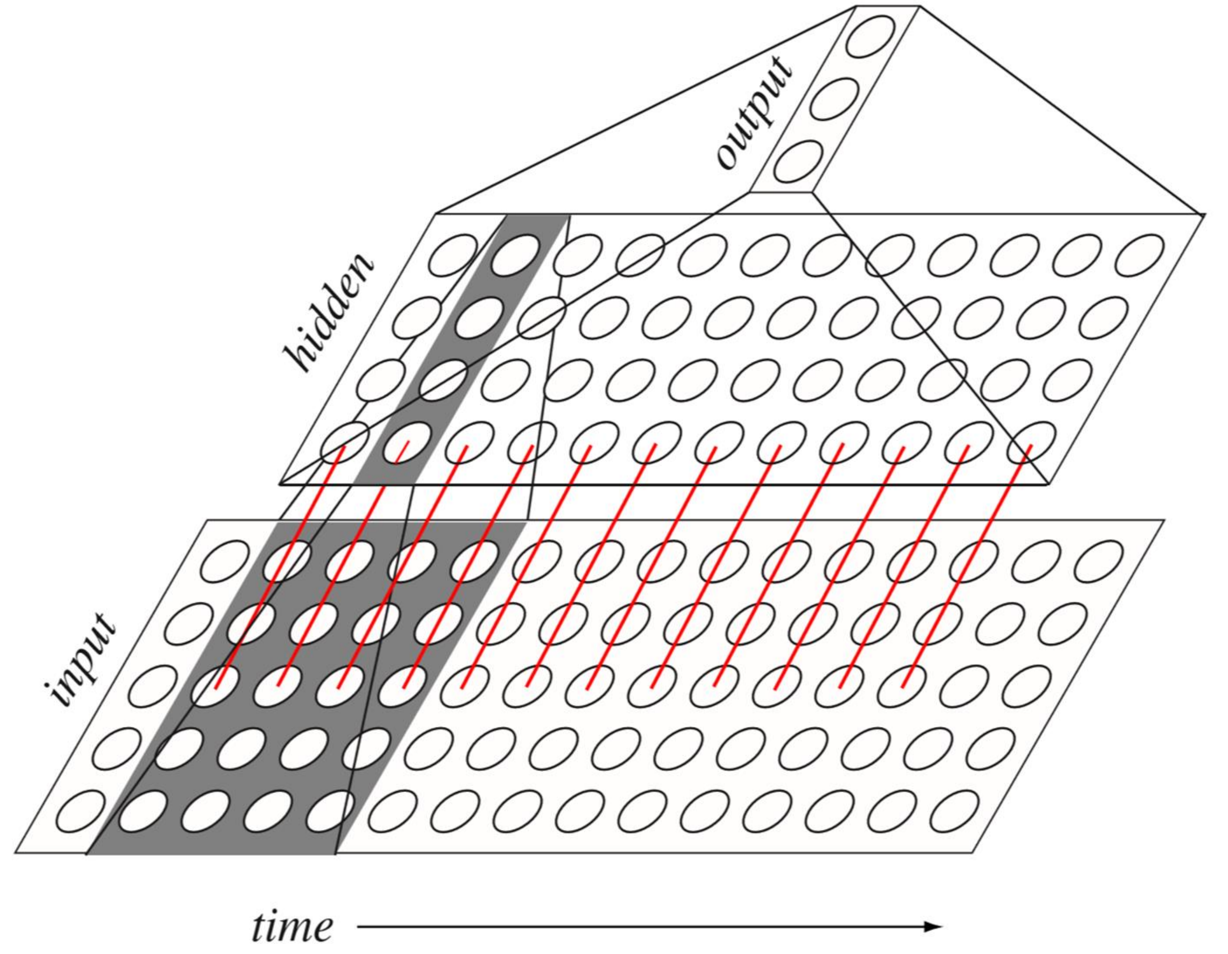
Practical Aspects of Backpropagation

- Number of Hidden Layers
  - Depends on complexity of classification problem
  - Unnecessary layers can cause minimization to caught into local minima

- Criterion Function
  - Entropy based information theoretic error functions
  - Minkowski error function (generalization of sum of squared error)

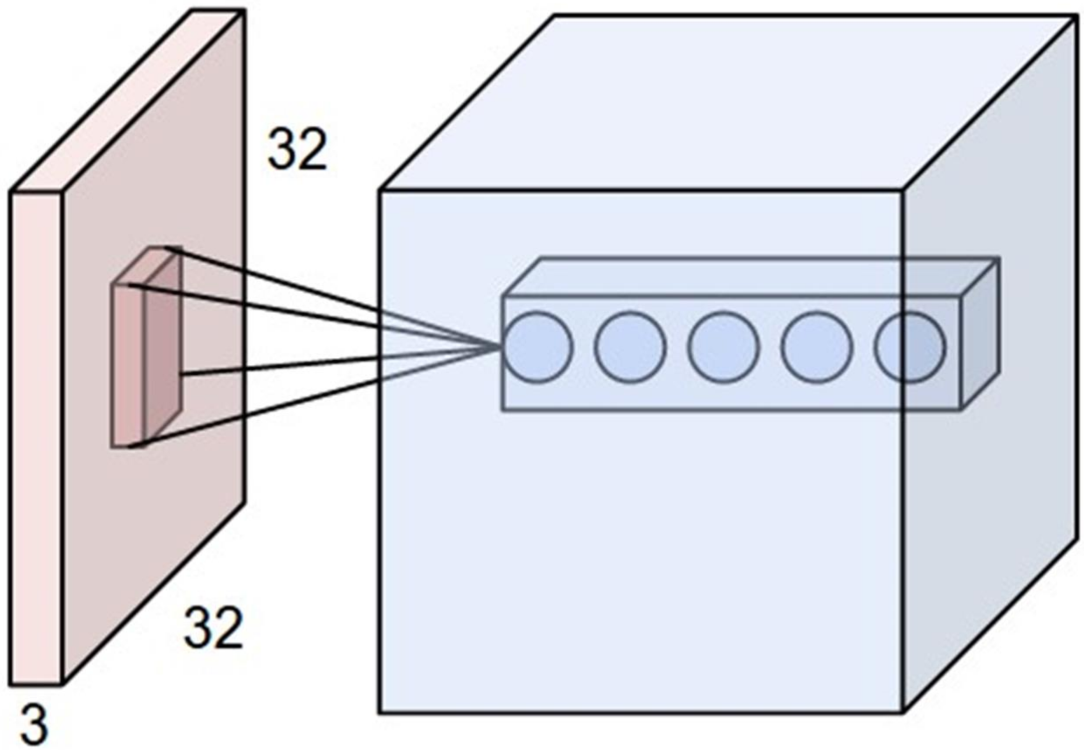
Convolution Networks

- More suitable for image data.



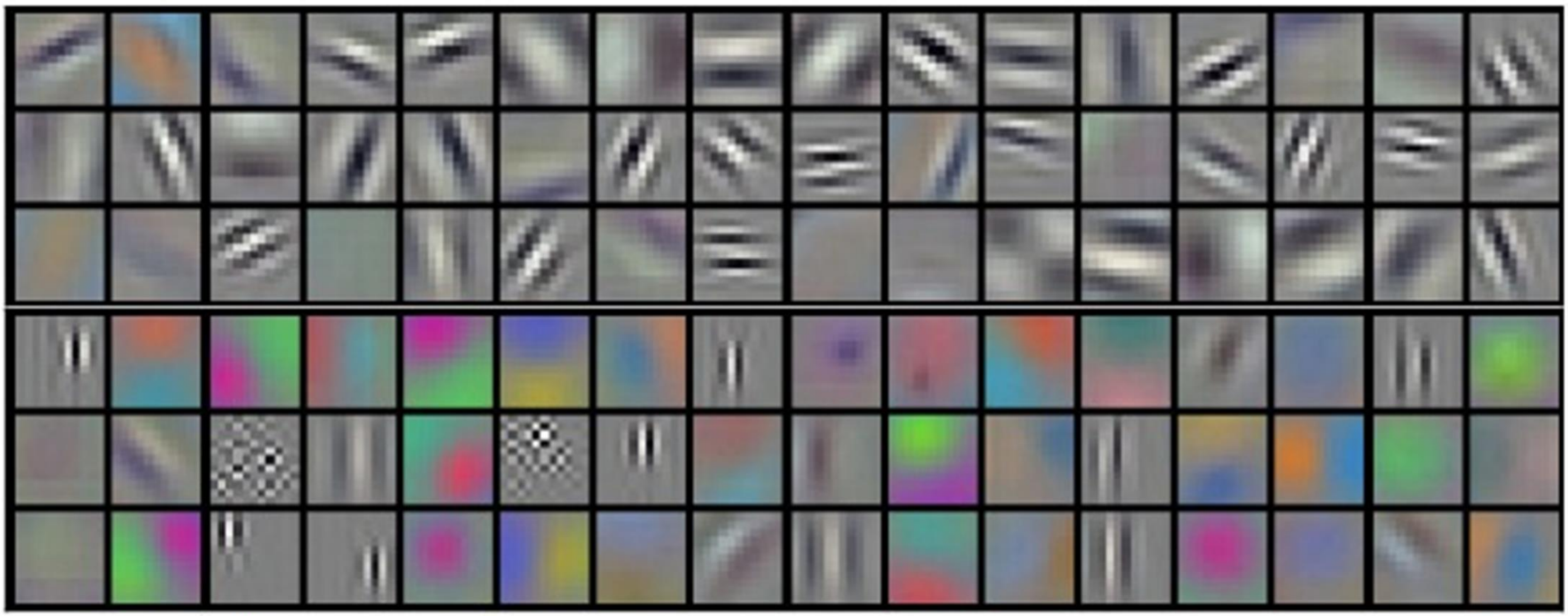
Convolution Networks

- http://cs231n.github.io/convolutional-networks/



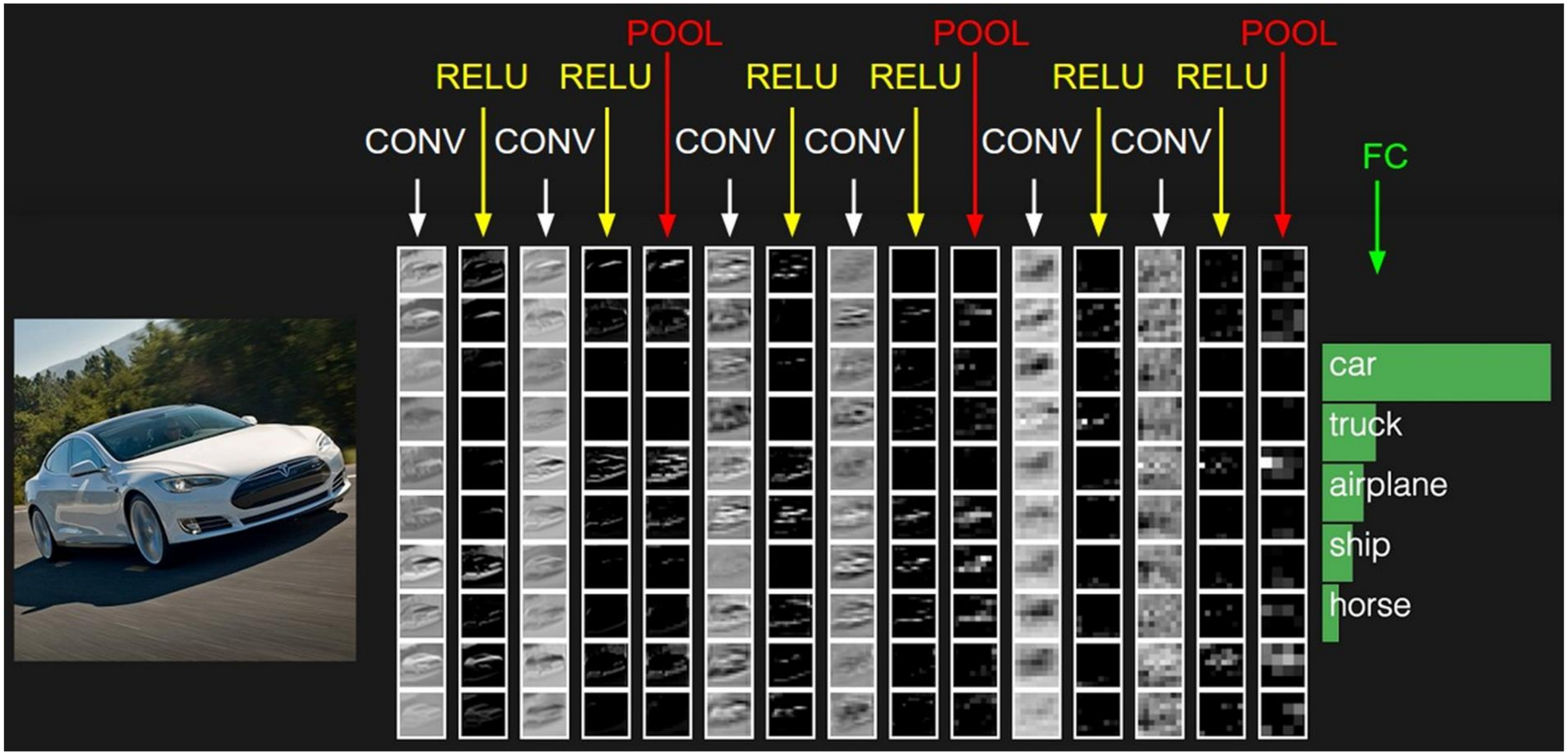
Convolution Networks

- http://cs231n.github.io/convolutional-networks/



Convolution Networks

- Live Demo at http://cs231n.stanford.edu/

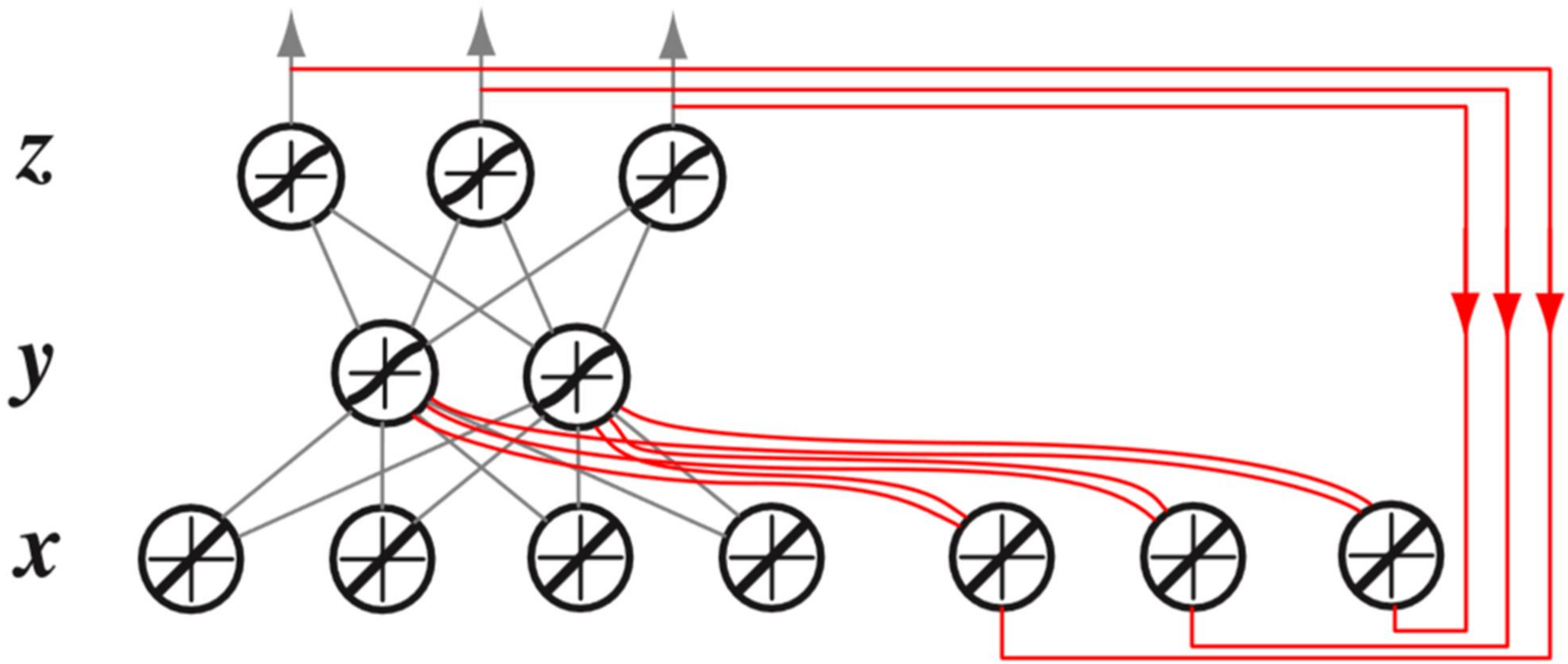


Convolution Networks

- **INPUT** [32x32x3] will hold the raw pixel values of the image,.
- **CONV** [32x32x12] layer will compute the output of neurons that are connected to local regions in the input, each computing a dot product between their weights and the region they are connected to in the input volume.
- **RELU** layer will apply an elementwise activation function
- **POOL** [16x16x12] layer will perform a downsampling operation along the spatial dimensions (width, height).
- **FC** (i.e. fully-connected) layer will compute the class scores, resulting in volume of size [1x1x10], where each of the 10 numbers correspond to a class score.

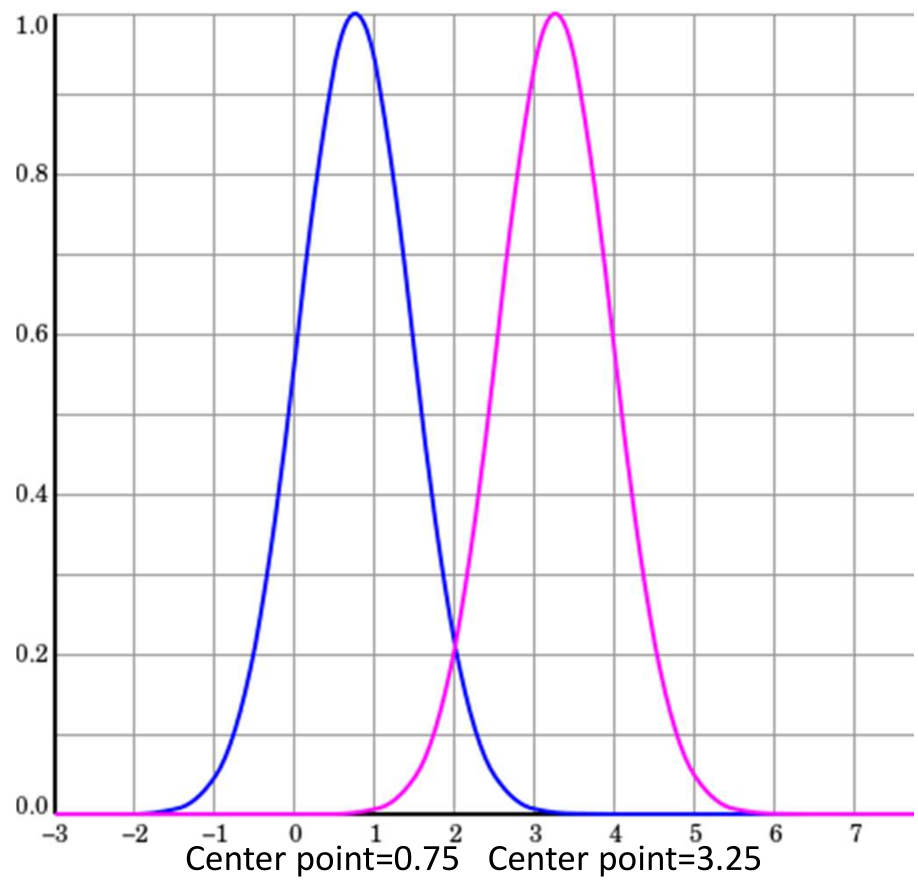
Recurrent Networks

- Useful for time-dependent signals with short periodic structures

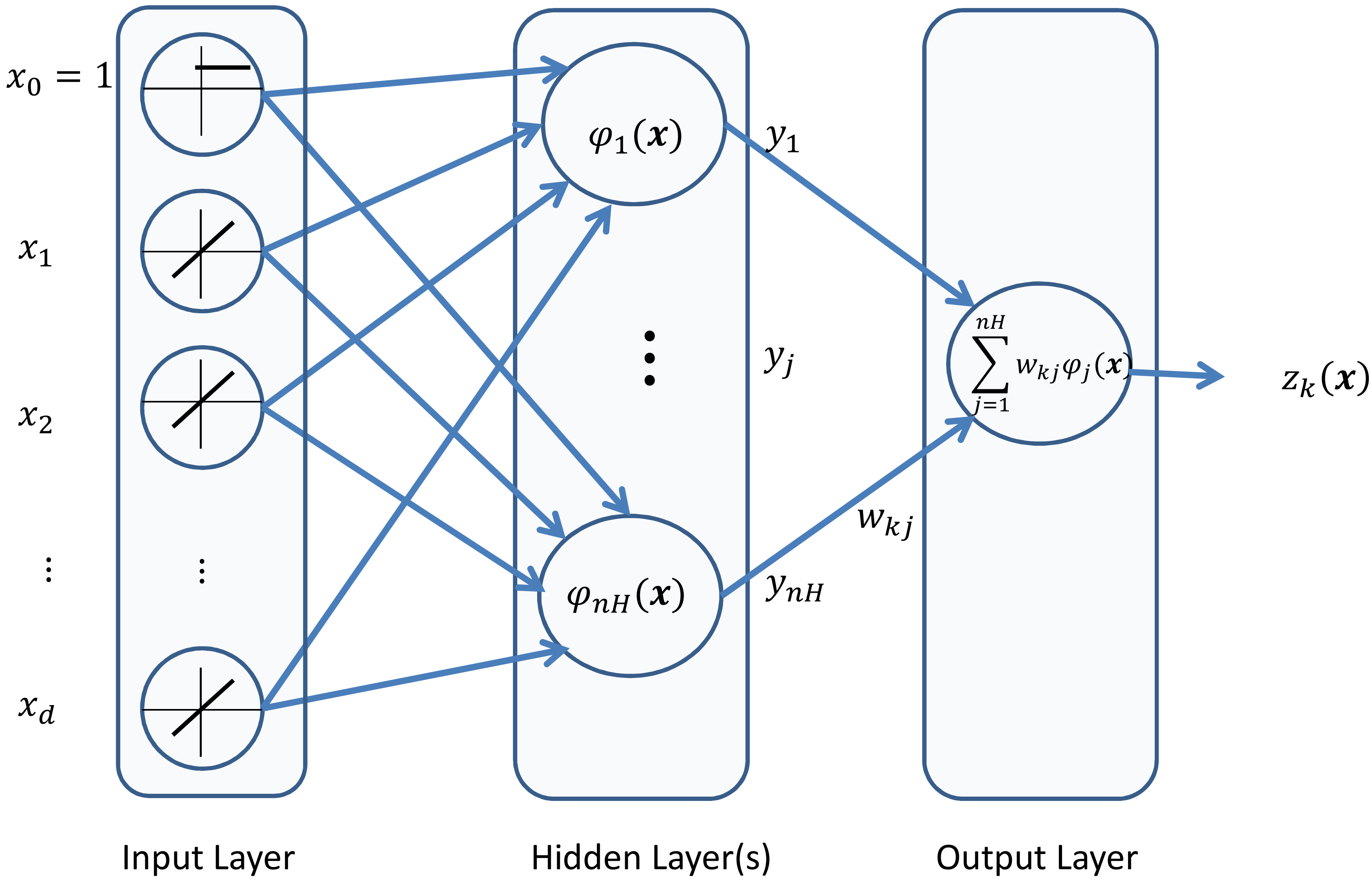


RBF Networks

- Radial functions are a special class of function
- Their characteristic feature is that their response decreases (or increases) monotonically with distance from a center point
- The center, the distance scale, and the precise shape of the radial function are parameters of the model.



RBF Networks



RBF Networks

- $z_k(\boldsymbol{x}) = \sum_{j=1}^{nH} w_{kj}\varphi_j(\boldsymbol{x})$
- Let $\boldsymbol{\Phi}_{m\times nH} = \begin{bmatrix} \varphi_1(\boldsymbol{x}_1) & \dots & \varphi_{nH}(\boldsymbol{x}_1) \\ \vdots & \dots & \vdots \\ \varphi_1(\boldsymbol{x}_m) & \dots & \varphi_{nH}(\boldsymbol{x}_m) \end{bmatrix}$
- Let $\boldsymbol{T}_{m\times 1} = [t_1 \quad \dots \quad t_m]^{\boldsymbol{T}}$
- $J(\boldsymbol{w}) = \frac{1}{2}\sum_{i=1}^{m}\|z_i - t_i\|^2$
- $\boldsymbol{\Phi}^{\boldsymbol{T}}\boldsymbol{\Phi}\boldsymbol{W}^{\boldsymbol{T}} = \boldsymbol{\Phi}^{\boldsymbol{T}}\boldsymbol{T}$
- $\boldsymbol{W}^{\boldsymbol{T}} = \left(\boldsymbol{\Phi}^{\boldsymbol{T}}\boldsymbol{\Phi}\right)^{-\boldsymbol{1}}\boldsymbol{\Phi}^{\boldsymbol{T}}\boldsymbol{T} = \boldsymbol{\Phi}^{\dagger}\boldsymbol{T}$
- BP can be used if $z_k(\boldsymbol{x}) = f\left(\sum_{j=1}^{nH} w_{kj}\varphi_j(\boldsymbol{x})\right)$ for any non-linear f