BITS Pilani

Machine Learning (IS ZC464) Session 11: Feed forward Neural Networks – Multilayer Perceptron (MLP) and Radial Basis Function Neural Network (RBFNN)

Computing Gradient

Revision

$E(w) = \sum_i T_i^2$

$= \sum_i (y^i - g(h_w(x^i)))^2$

where T_i is the error term for the i^{th} observation and is given by the difference between the desired output (y^i) value and the estimated value ($h_w(x^i)$)of the output

$T_i = y^i - g(h_w(x^i))$

$h_w(x^i)$ is the hypothesis function given by

$h_w(x^i) = w_1 x^i_1 + w_2 x^i_2 + \ldots . w_n x^i_n$

Observe: E is a function of w.

Note: Here the superscript 'i' represents the 'i'th observation and NOT the power of x.

Observe

- E is the function of T_i

Revision

- Ti is the function of g (assuming y as constant)
- g is the function of h
- h is the function of w

- Chain rule of Differentiation

$$\partial E/\partial w_k = \sum_i \partial E/\partial T_i * \partial T_i/\partial g * \partial g/\partial h * \partial h/\partial w_k$$

Equation 1

Observe

Revision

Since

$$E(w) = \sum_i T_i^2$$

$$\partial E/\partial T_i = 2*T_i$$

Chain rule of Differentiation

$$\partial E/\partial w_k = 2*\sum_i T_i * \partial T_i/\partial g * \partial g/\partial h * \partial h/\partial w_k$$

Also since

Equation 2

$$T_i = y^i - g(h_w(x^i))$$

Therefore

$$\partial T_i/\partial g = 0 - 1 = -1$$

Working with derivatives

Revision

Equation 2 now becomes

$$\partial E/\partial w_k = 2 * \sum_i T_i * (-1) * \partial g/\partial h * \partial h/\partial w_k$$

Also since

$$\partial g/\partial h = \partial\, g(h_w(x^i))\, /\partial h = g'$$

Equation 3

And

$$h_w(x^i) = w_1 x^i_1 + w_2 x^i_2 + \ldots . w_n x^i_n$$

Therefore

$$\partial h/\partial w_k = x^i_k$$

Hence equation 3 is simplified as

$$\partial E/\partial w_k = -\, 2 * \sum_i T_i * g' * x^i_k$$

Equation 4

Computing gradient in the direction of w_k

Revision

- Substitute expression for T_i in equation 4

$$\partial E/\partial w_k = -\ 2 * \sum_i (y^i - g(h_w(x^i))) * g' * x^i_k$$

Equation 5

- The Weight update in the direction of w_k

$$\Delta w_k = -\ 2 * \sum_i (y^i - g(h_w(x^i))) * g' * x^i_k$$

Equation 6

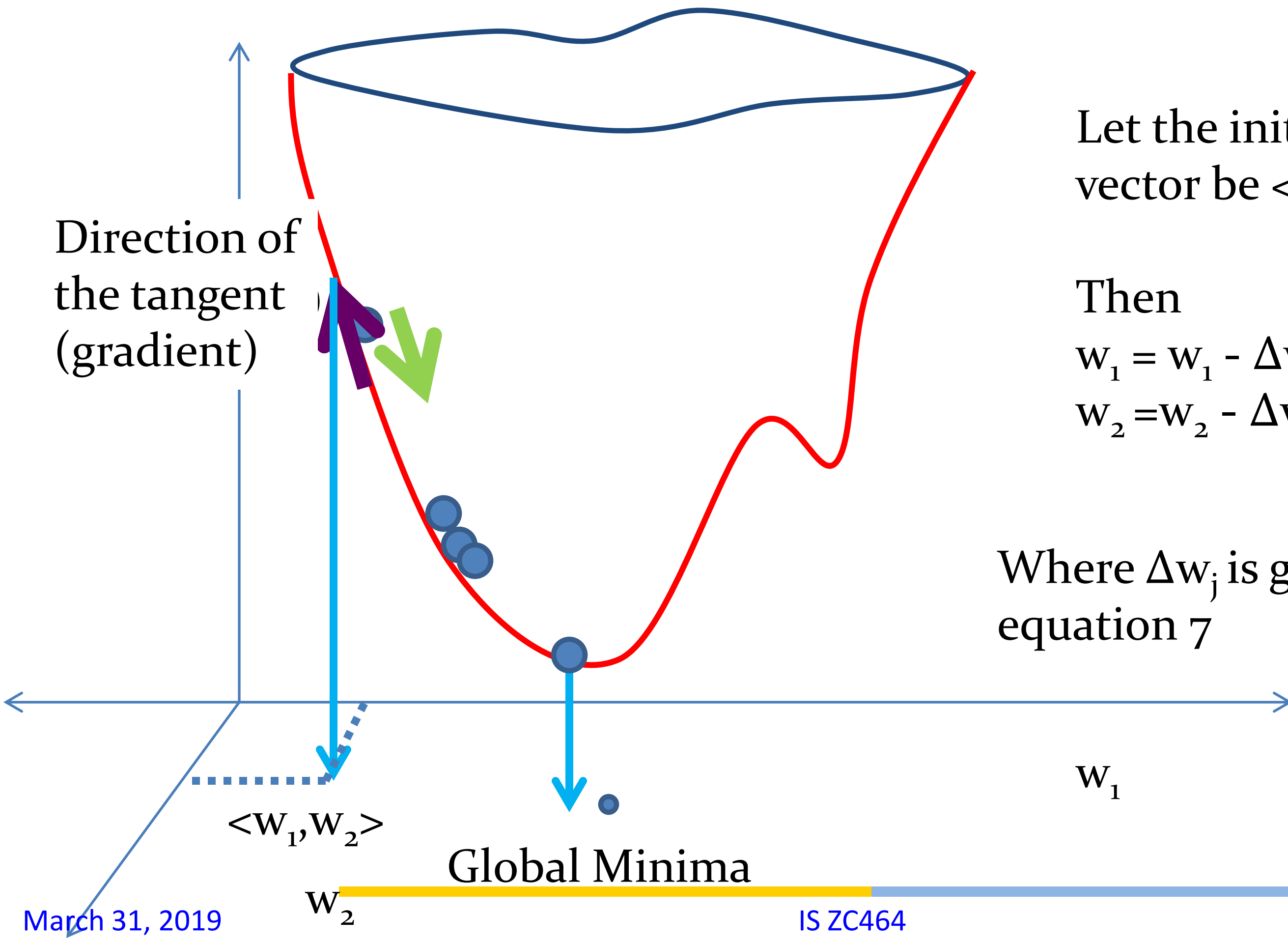
Where 2 can be dropped to bring normalization.

$$\Delta w_k = -\ \sum_i (y^i - g(h_w(x^i))) * g' * x^i_k$$

Equation 7

Delta Learning: Modification of the Initial weight

Revision



Let the initial weight vector be <w_1,w_2>

Then

$w_1 = w_1 - \Delta w_1$

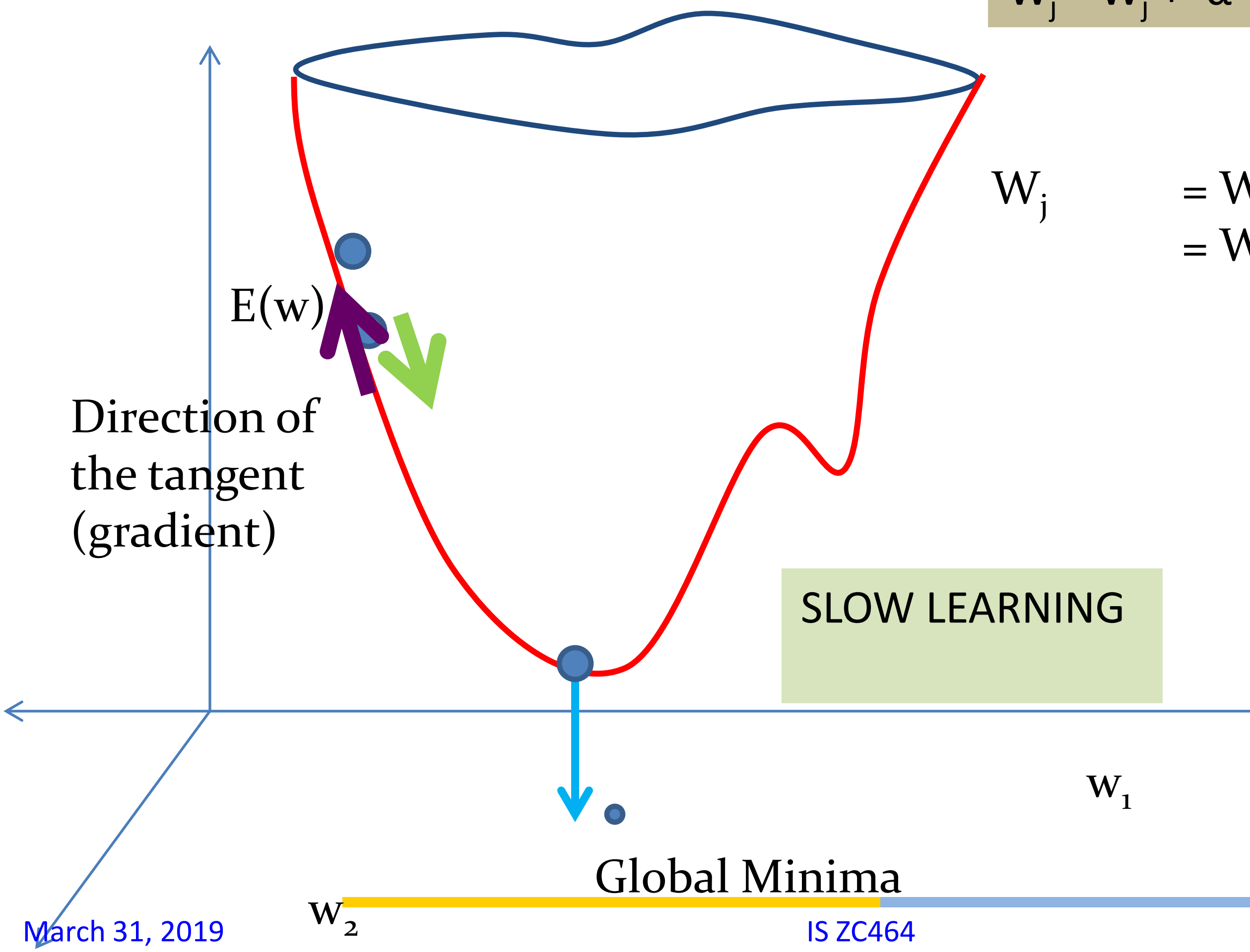
$w_2 = w_2 - \Delta w_2$

Where Δw_j is given by equation 7

Learning rate: fast or slow learning

Revision

$$W_j = W_j + \alpha * \sum_i (y^i - g(h_w(x^i)))_* \, g'_* \, x^i_k$$



$$W_j = W_j - \Delta w_j$$
$$= W_j + \sum_i T_{i*} \, g'_* \, x^i_k$$

α

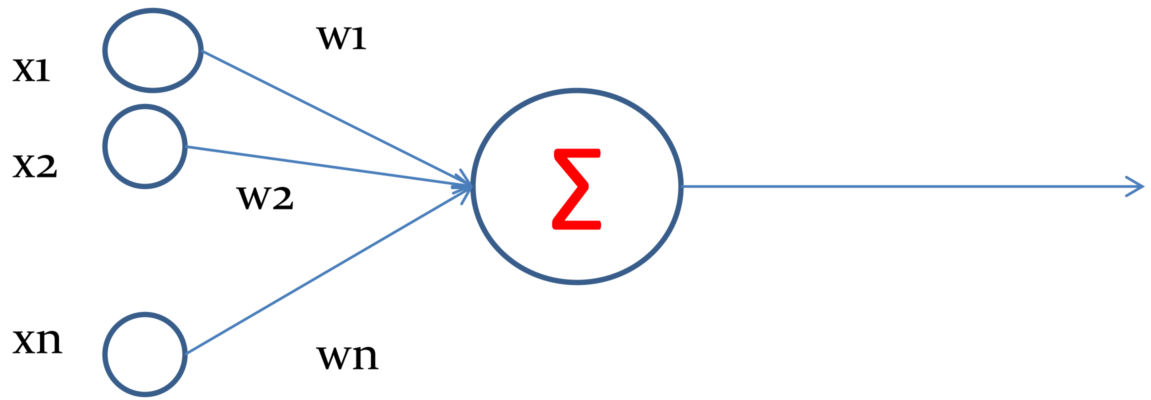
α ranges from 0 to 1
If it is 0 , there is no learning
If it is large the learning rate is high

Feed Forward and Back propagation

- In feed forward neural network, the weights are computed on the basis of the input propagating through neurons in the forward direction. In this no neuron receives the modified input.
- In back propagation neural network, the processed input is cycled again through the previous layer neurons and the weights are modified.

Feed Forward Neural Networks

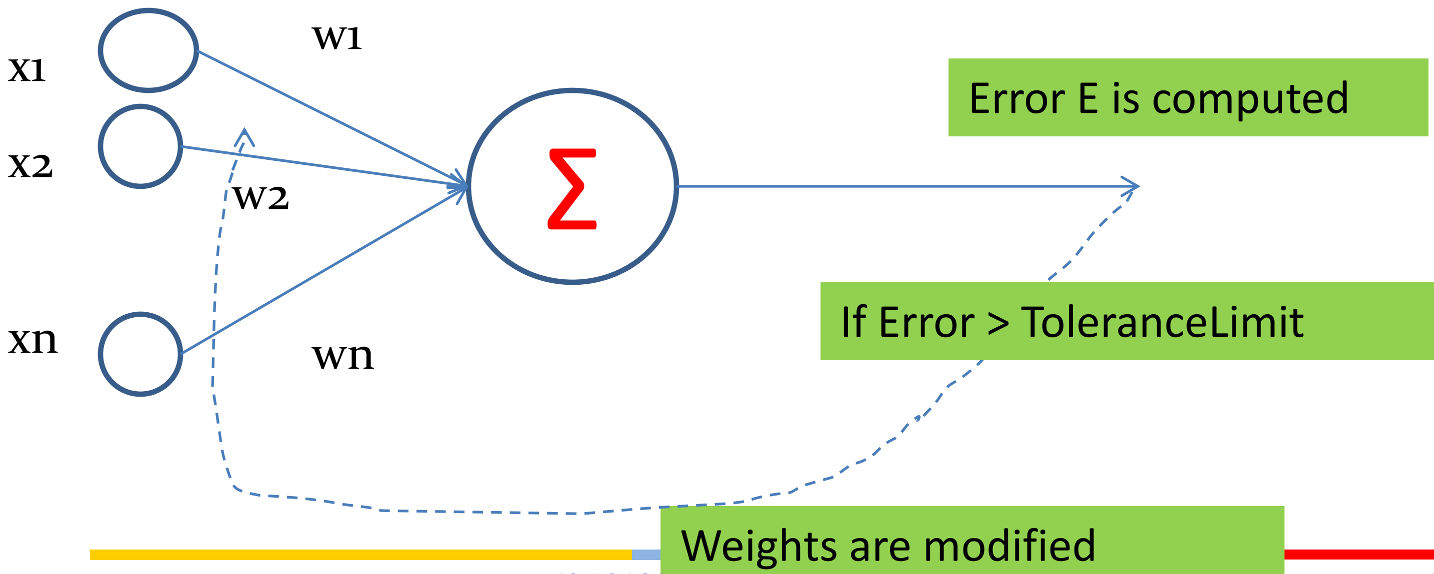
Weights learning is one way



Back Propagation Neural Networks

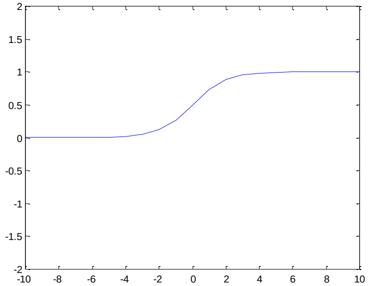
Weights learning is cyclic

If Error <= ToleranceLimit
Then
terminate



Activation function should be differentiable for Delta Learning

- $\Delta w_k = - \sum_i (y^i - g(h_w(x^i))) * g' * x^i_k$
- The function g' is 0 if g is not differentiable
- Example Activation functions
- Step Function : Not differentiable
- Sigmoid Function : Differentiable



$$y = \frac{1}{1 + \exp(-x)}$$

Gradient Descent Algorithm

1. Initialize weights in the n-dimensional space randomly.
2. Compute error E.
3. Define error tolerance limit L.
4. While (E > L)
   - Modify weights W according to delta rule
   - Compute error E with the modified weights and the given input.

Terminology used in text book

	Used in the slides here	Used in book by Mitchell (Chapter 4)
Set of Training samples	Input : vector x^i : i = 1,2,…m output: y^i : i = 1,2,…m	D is the set of training samples $d \in D$ Input : vector x_d : $d \in D$ output : t_d : $d \in D$
Target (Known-supervised)	y^i	t_d
Input feature vector	$x^i = <x^i_1, x^i_2, x^i_3, \ldots x^i_n>$	$x_d = <x_{d1}, x_{d2}, x_{d3}, \ldots x_{dn}>$
Output-predicted by ANN	$h_w(x^i)$	o_d
error	$y^i - h_w(x^i)$	$t_d - o_d$

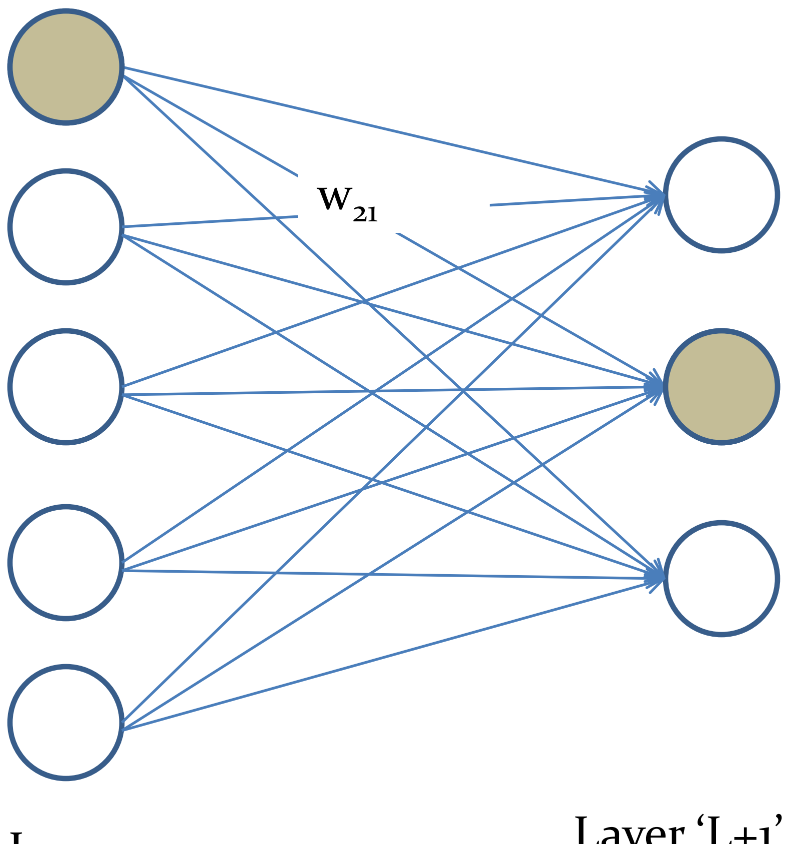
What to do with the weights (W) obtained using Gradient Descent Algorithm

- Let W = $<w_1, w_2, w_3, \ldots w_n>$ [as a result of training]
- Have a new feature vector is $x = (x_1, x_2, x_3, \ldots x_n)$ corresponding to the sample not yet seen by the machine (known as test vector)
- Compute output y as follows
- $y = h_w(x) = w_1x_1 + w_2x_2 + \ldots w_nx_n$
- This is the identification of the output **[Machine has learned]**

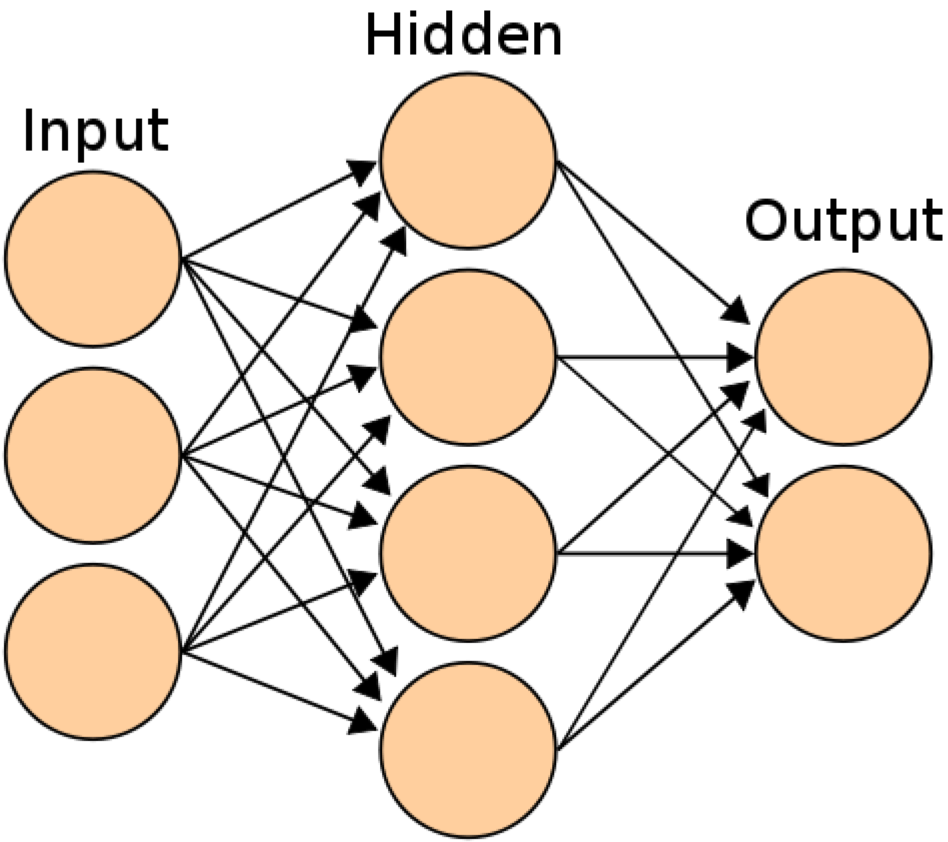
Multilayer Feed Forward neural network

- These represent the class of networks which approximate the complex functions.
- The network has one or more hidden layers.
- The neuron ‘i’ of layer ‘L’ is connected by a synaptic weight w_{ki} to the ‘k’th neuron of layer ‘L+1’

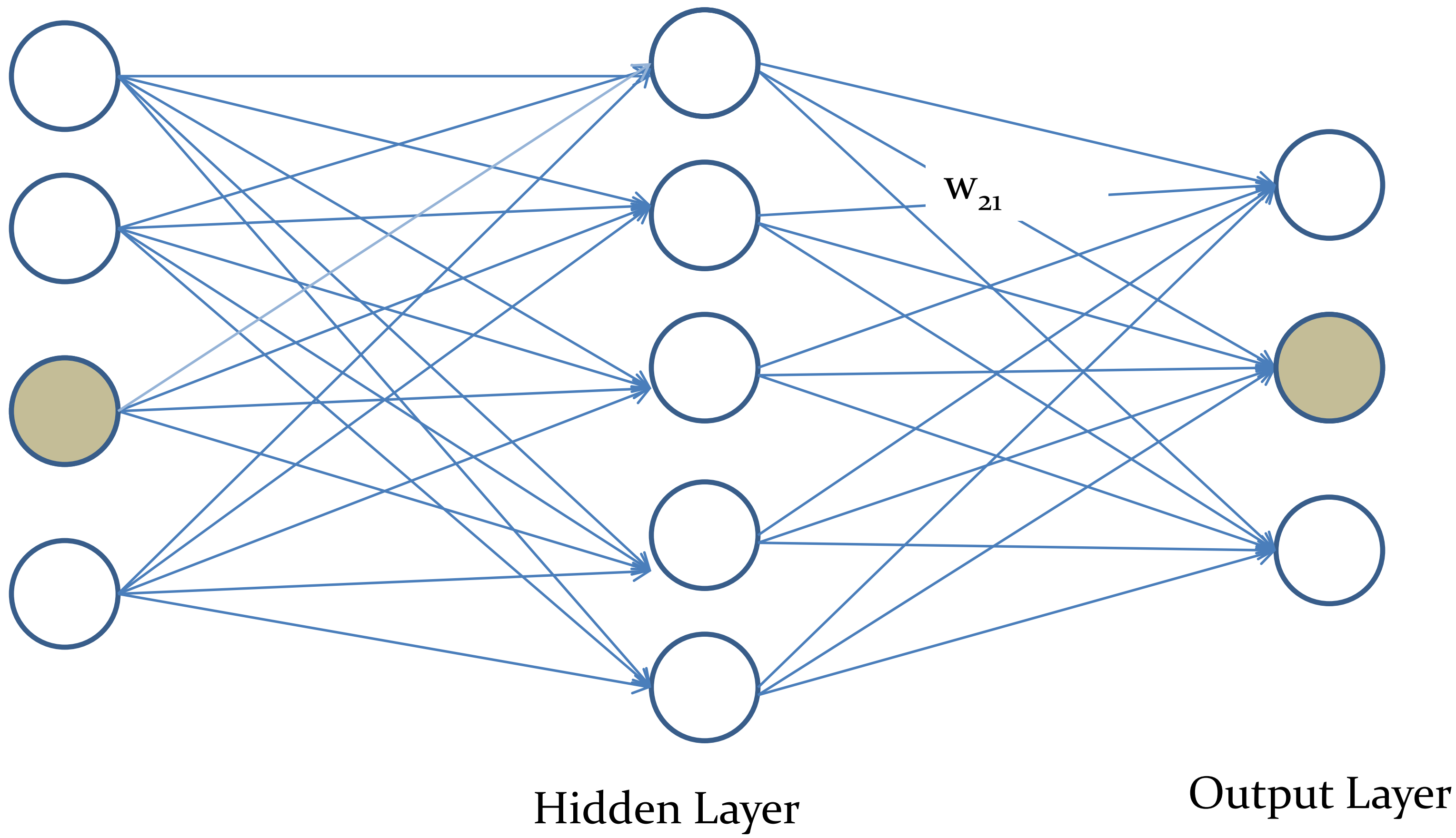
Weight Terminology



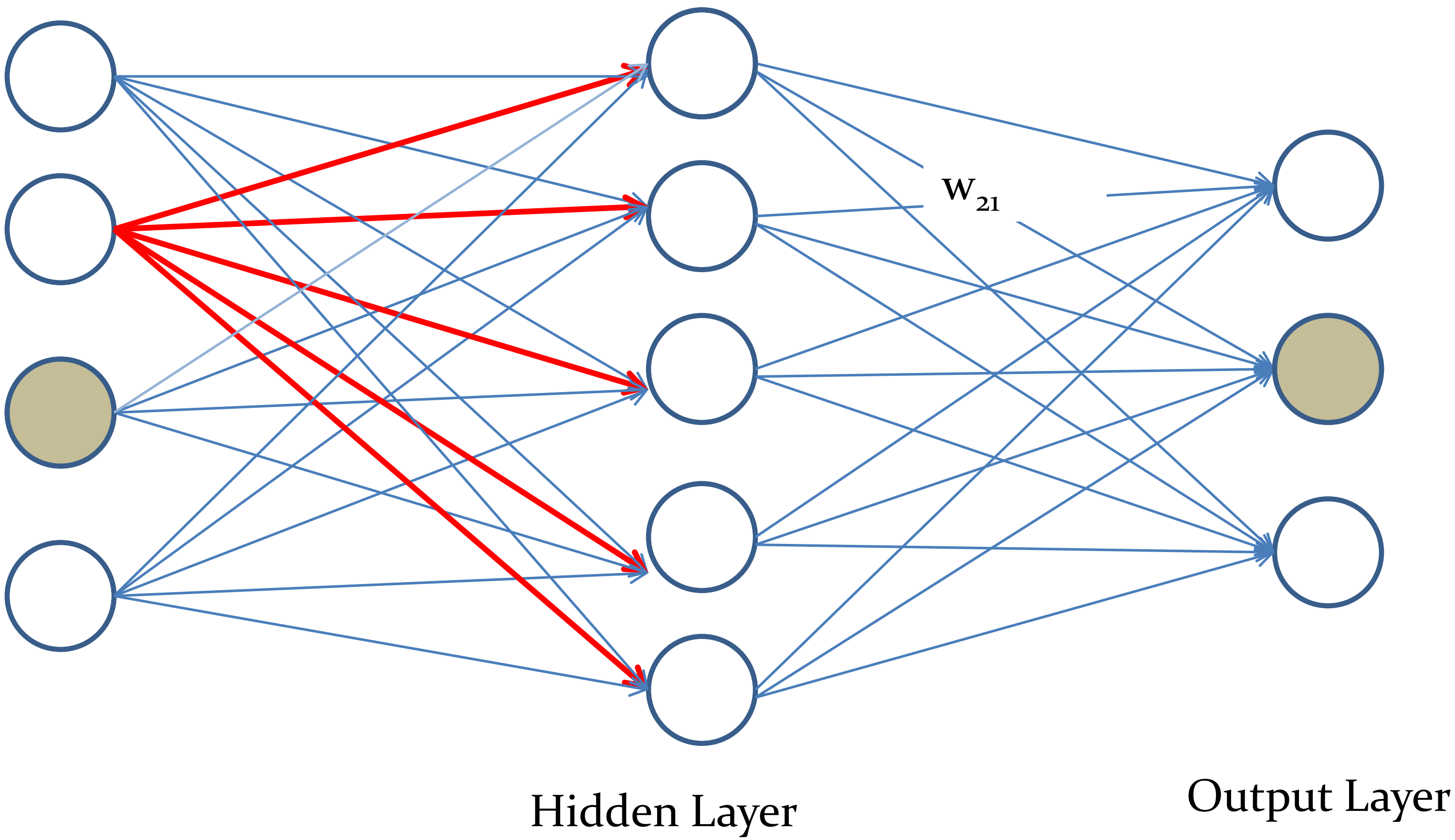
MLP



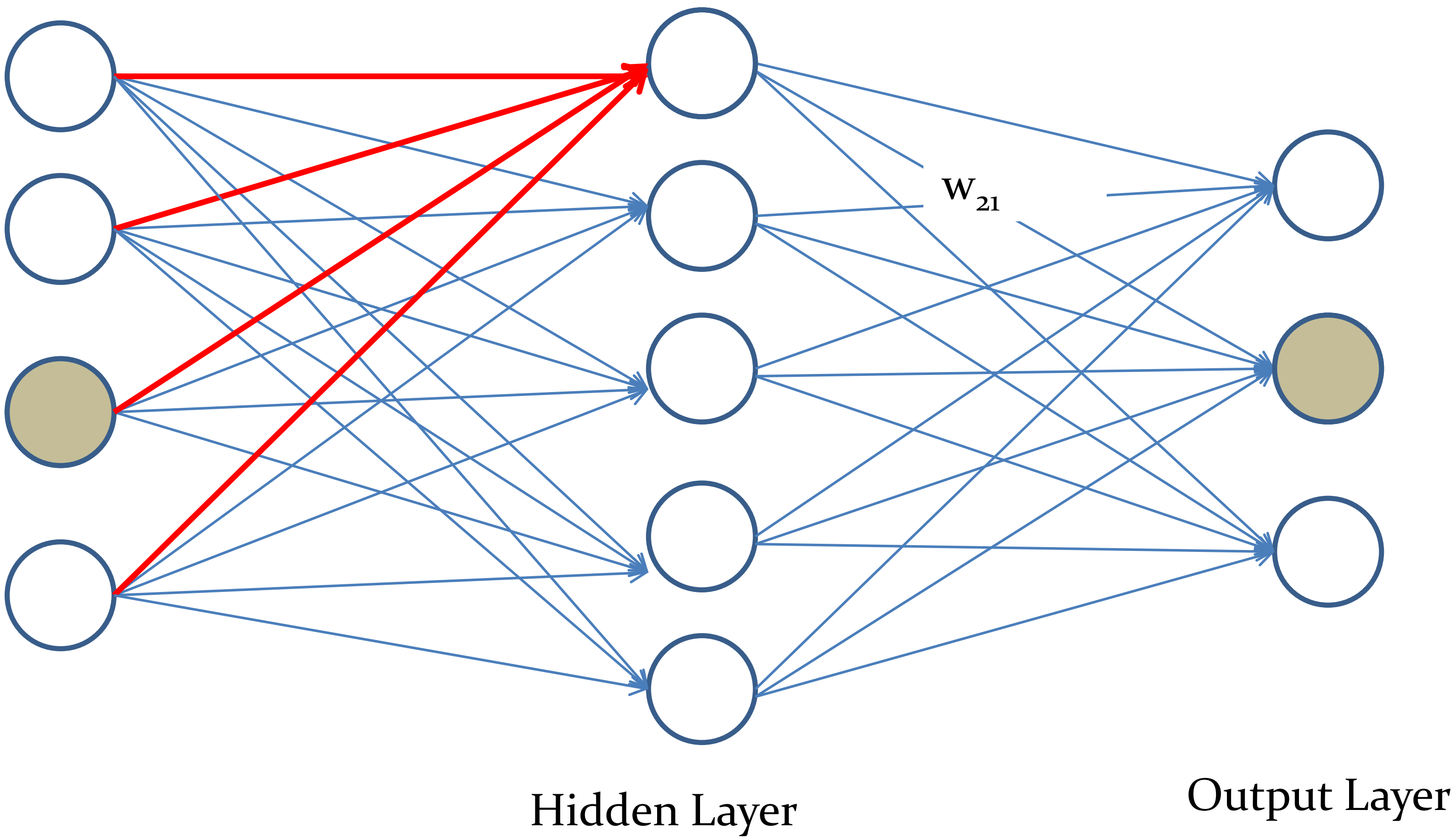
Multilayer Feed forward Neural Network



Multilayer Feed forward Neural Network



Multilayer Feed forward Neural Network



Multi Layer Perceptrons

- These are acyclic directed graphs.
- MLP is a feedforward neural network.
- Can handle non linearly separable data.
- Have different hidden layers of neurons which process the data.
- Training is through weight learning.
- i^{th} layer passes information to $i+1^{th}$ layer

Real World Problem

- Face Recognition

A face to recognize …….for a Computer

… for Humans

196	203	204	202	202	202	205	206	209	207	206	207	205	203	201	196	189	180	168	150
199	192	181	168	164	164	163	166	172	183	193	195	197	198	200	200	197	189		
177	175	178	167	145	127	117	120	123	131	145	163	178	189	196	196	193	187		
179	194	197	188	175	159	138	127	125	121	123	138	156	176	190	189	181	172	159	146
192	194	186	171	162	159	153	145	140	139	138	142	156	178	191	185	170	152	132	123
186	176	154	151	164	162	152	136	127	135	140	146	166	191	198	190	171	145	124	119
181	165	158	161														136	116	113
183	159	127	127														125	105	107
192	163	126	149														121	103	110
202	193	170	164	161	142	118	117	127	130	147	169	193	203	207	199	163	131	117	122
205	202	192																	137
204	202	199																	154
206	206	205																	165
206	207	207																	162
206	207	207	207	207	205	205	205	204	205	204	204	207	207	203	190	165	146	142	157
207	206	206	207	207	204	199	198	200	200	203	206	207	206	204	195	173	149	134	144
207	205	205	205	205	202	191	187	188						200	192	168	145	140	134
204	205	205	204	203	200	191	183	185						191	171	146	136	144	135
								192						162	136	111	130	137	132
								199						136	111	81	110	129	136
								200						129	116	107	114	131	139
								203								131	133	139	141
197	198	197	194	195	197	200	203	204								139	143	144	145
197					199	201	205	203	202	202	198	193	178			142	145	146	148

Patterns…

Whose face is this ?
Machine can recognize if trained..

Which pattern of numeric values represents a person’s face uniquely?

Human’s Face Recognition ability is amazing…….

?

A

B

C

A Face Image

- It is simply a grid of numeric values for the machine.
- A machine uses its computational powers to identify patterns from the above numeric values. (Feature Extraction)
- These patterns are unique to a person.
- A face image is represented by various numerical ways such as PCA eigen faces, DCT, wavelets, other statistical methods.

Understanding Patterns and Pattern Recognition Problem

Training Patterns

Class 1: <1, 2, 3, >

Consecutive integers in ascending order

Class 2: <1,4,9>

Squares of Consecutive integers

Class 3: <-1, -3, -5>

Descending integers with step size 2

Testing Pattern : <25, 36, 49>

Humans: Recognize easily (Good Generalization Capability)

Machines: Need Mathematical Models to recognize patterns

Patterns

- Individual values in the pattern do not give valuable information about the pattern.
- All values in association with each other are informative.
- Patterns have an underlying mathematical structure.

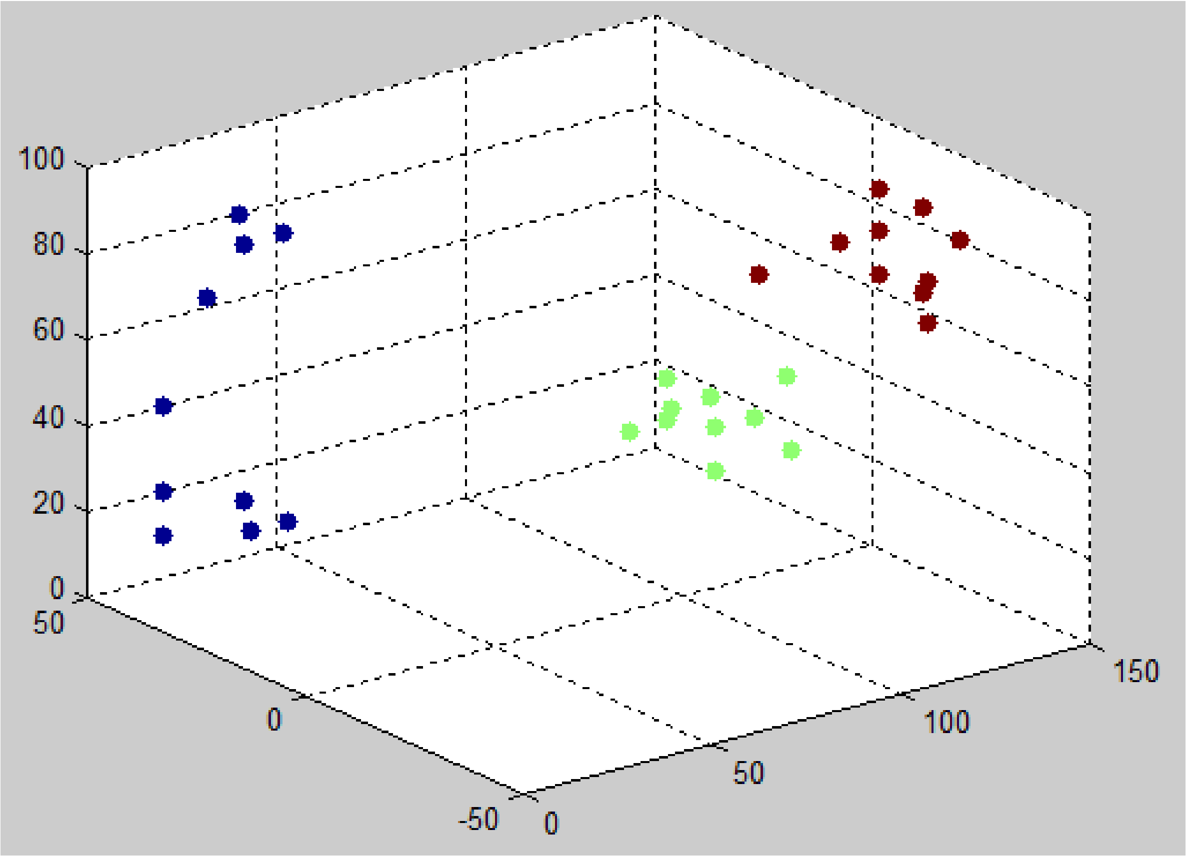
Complexity of Face Data

- The geometric face features are not robust with respect to variations in expression or illumination conditions.
- Mathematical representations such as coefficients of the Discrete Cosine Transform, Wavelet Transforms etc. are used to represent the face.

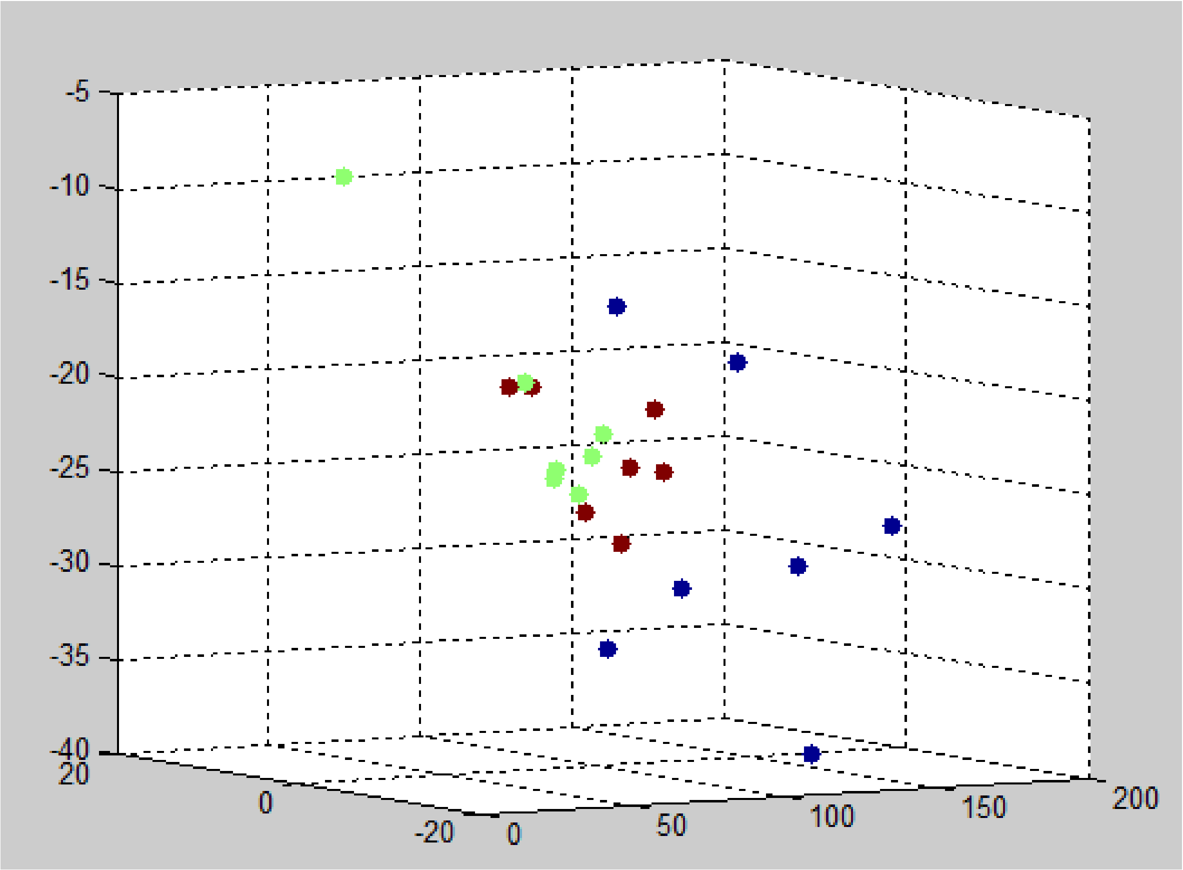
Complexity of Face Data

- A large number of such coefficients are required to retain identity of a person face.
- A small number of the Optimal features are selected. (to reduce computational load)
- The number (n) of optimal features is also high (e.g. 45 as against all 10000 pixels)

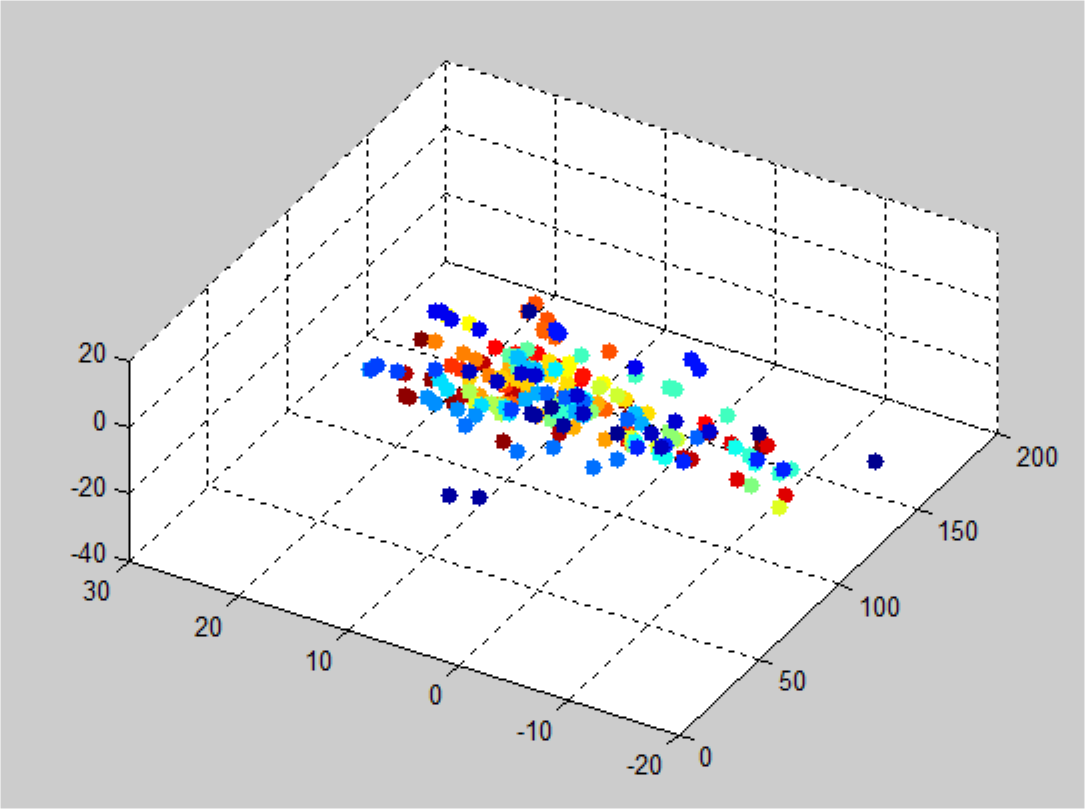
Linearly Separable Non-Face Data



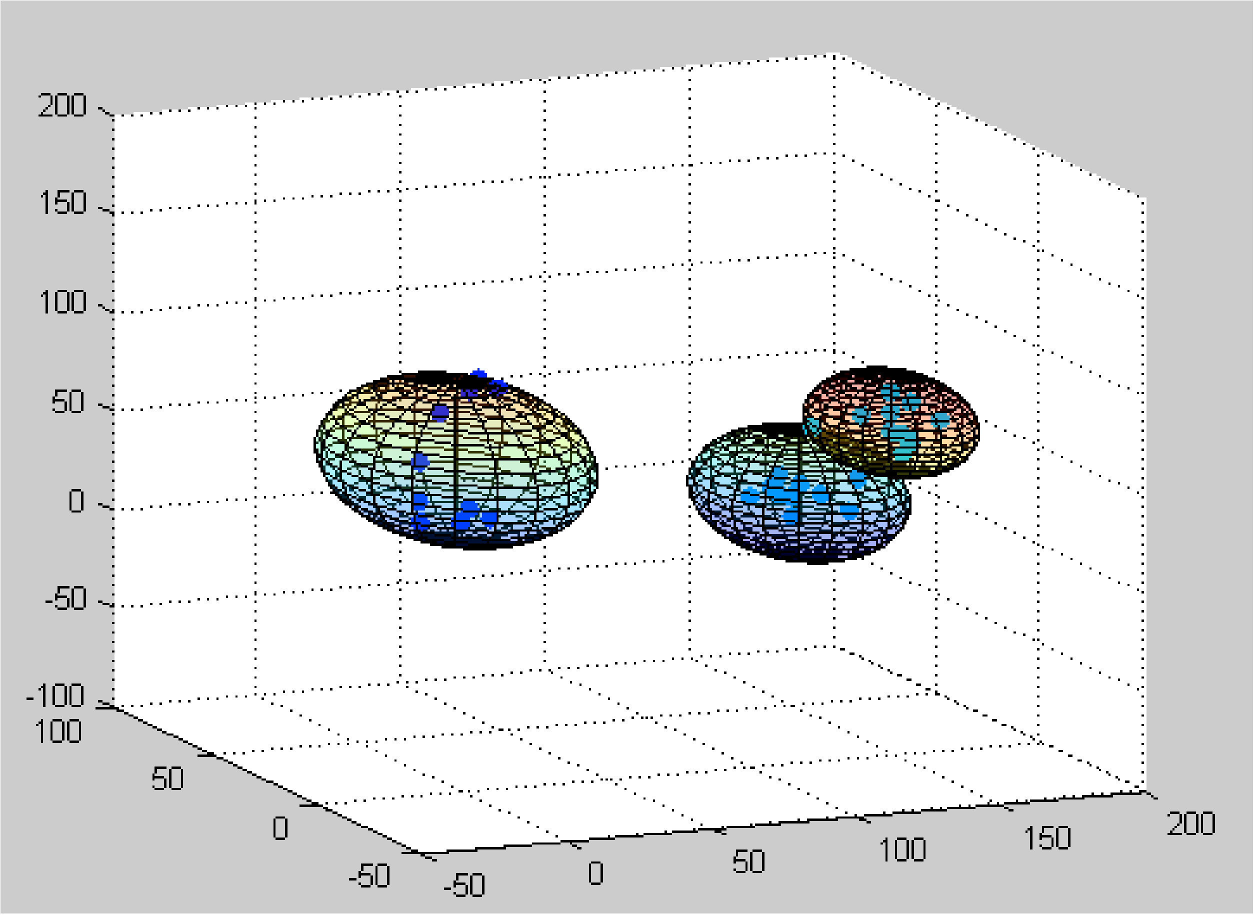
Each face is a point in the n-dimensional space. (ORL face data for three persons)



The points in the n-dimensional space cannot be clustered (colorwise) by hyperplanes.

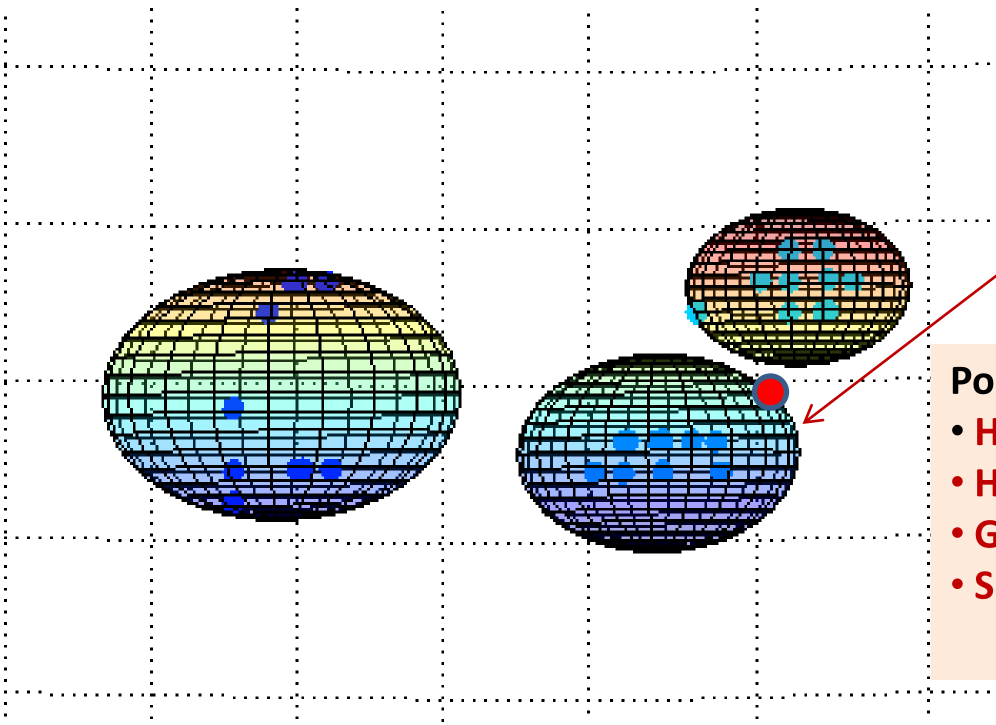


Face data is nonlinearly separable (Hyper-Surfaces can create boundaries between clusters)



Classification Problem

Given Training Data



Closest cluster to the n-dimensional test feature vector is computed

Possible Decision Boundaries

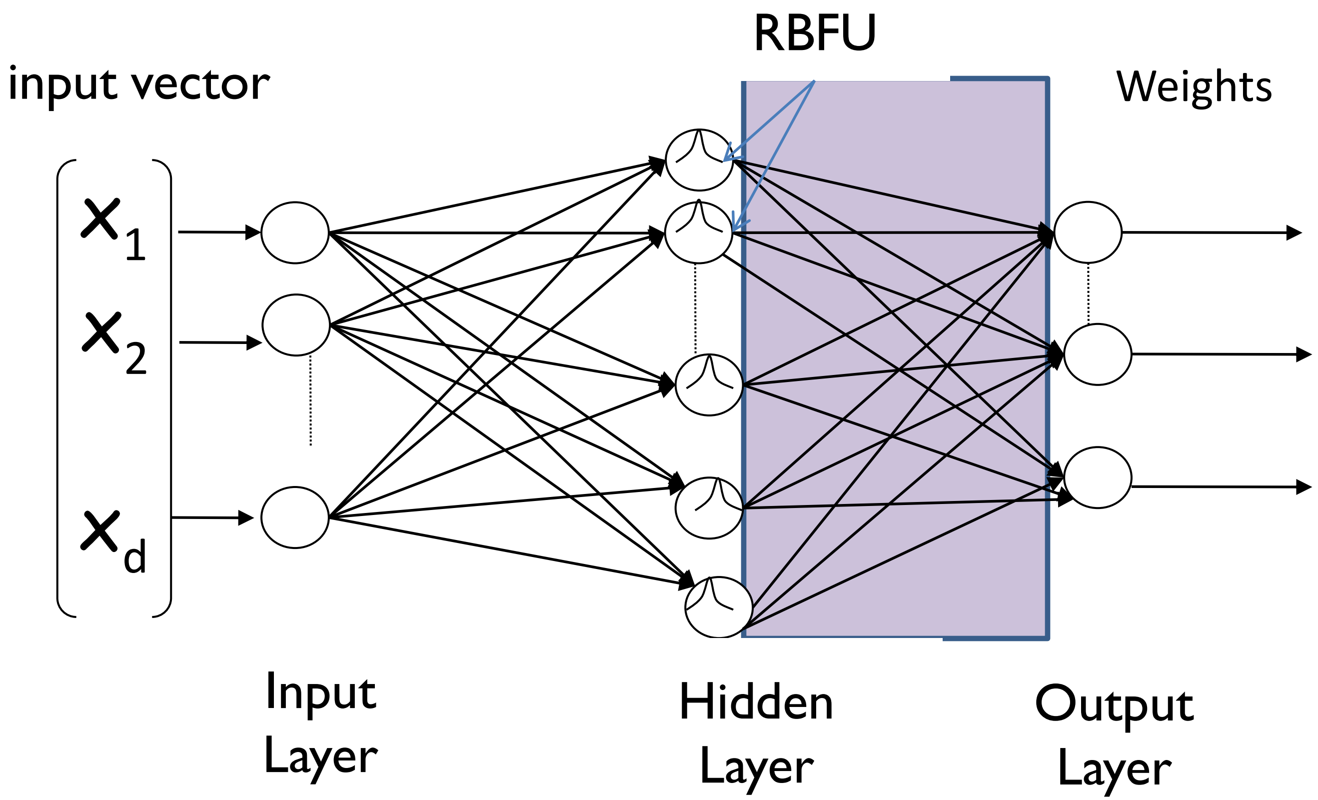
- **Hyper Plane**
- **Hyper Sphere**
- **Gaussian Surface**
- **Support Vectors**

Challenge: Design of Decision Boundary

Face Recognition Problem

- Posed as a classification problem
- Classes are the person names (identity)
- Training face images are visualized as points in d-dimensional space (d: pattern size)
- Challenge is in identifying appropriate boundaries demarcating individual cluster.

Radial Basis Functions Neural Network (RBFNN) Architecture



Why More neurons?

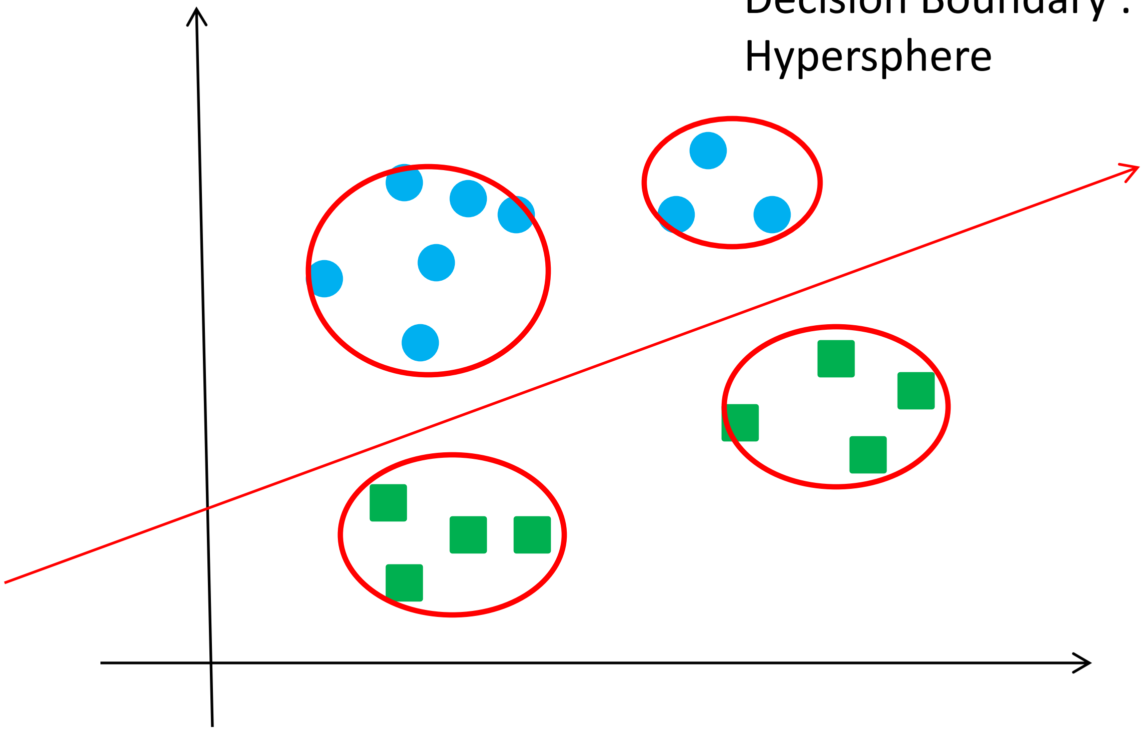
- More number of classes.
- Large input sizes of the patterns
- Nonlinear separability of clusters in n-dimensional space.

RBFNN

- They capture the training environment in terms of weights.
- The radial basis functions units (RBFU) locally capture the structure of the data
- Basis functions at the RBFU play an important role in transforming the nonlinearly separable high dimensional data to a space of linearly separable data.

Multi Layer Perceptron Vs. RBFNN

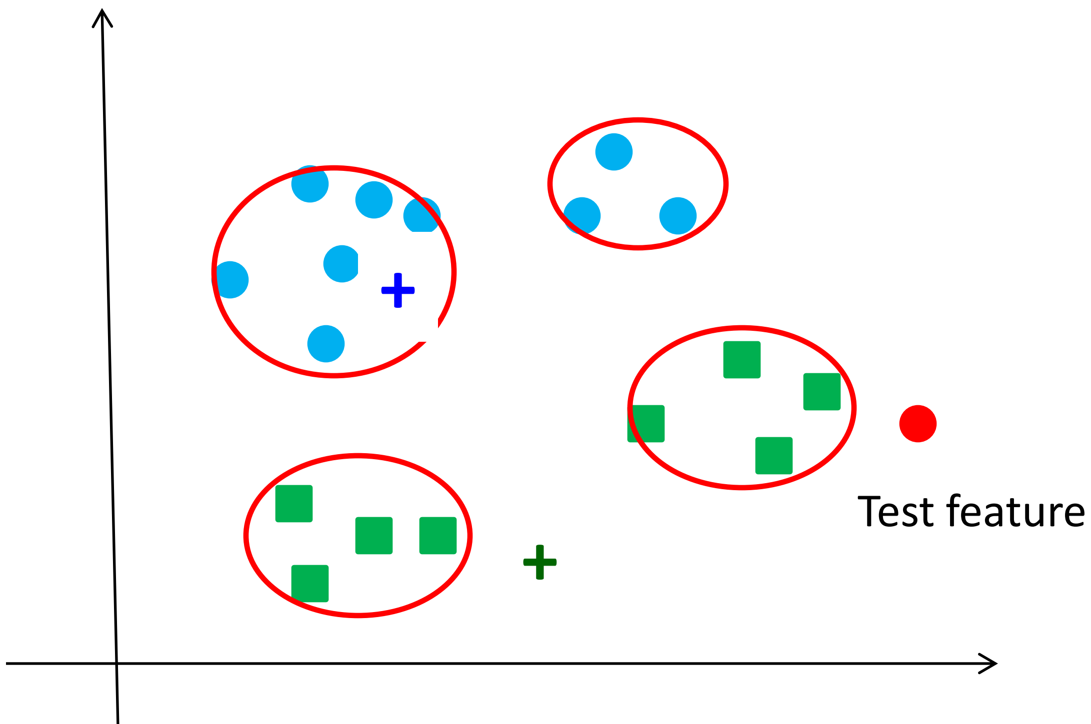
Decision Boundary : Hypersphere



The center of the natural cluster is the center of the hidden neuron

A hidden neuron is sensitive for data points near its center

Nearest Neighbor Classification Vs. RBFNN based classification



“Do not know “ condition can be handled well by RBFNN

Nearest neighbor: Shortest distance to the mean of the cluster

RBFNN: Within limits of Radial distance to the mean of the cluster