# Approximation of functions

A multi layer perceptron trained through BPA can be an engine for learning the non-linear I/o mappings

$$f : \mathbb{R}^{m_0} \longrightarrow \mathbb{R}^{m_L}$$

Space of I/p neurons

Space of o/p neurons

Qn: What is the min # of hidden layers needed for a MLP to learn the I/o mapping?

# Universal Approximation Theorem (Key: 1 single layer of hidden neurons will suffice)

Let $\varphi(\cdot)$ be a non constant, bounded and monotone non-decreasing continuous function. Let $I_{m_0}$ denote a $m_0$-dim. unit hyper cube $[0,1]^{m_0}$. The space of cont. functions on $I_{m_0}$ is denoted by $C(I_{m_0})$. Then, given any function $f \in C(I_{m_0})$ and $\varepsilon > 0$ $\exists$ an integer $m_1$ & sets of constants $\alpha_i, b_i, w_{ij}$ $i = 1, \dots, m_1$ & $j = 1, \dots, m_0$ /

$$\underbrace{F(x_1 \dots x_{m_0})}_{\text{approximation}} = \sum_{i=1}^{m_1} \alpha_i \varphi\left( \sum_{j=1}^{m_0} w_{ij} x_j + b_i \right) \text{ is an}$$

approximation of the true $f(\cdot)$ / $\|F(\cdot) - f(\cdot)\|_p < \varepsilon$ ← typically $p = 2$

# Bounds on Approximation errors

Barron (1993) established the approx. properties of a MLP

Suppose $f$ is the fn. representing the data points

The n/w gives us the approx. function $\underbrace{F}_{}$.

$$F \sim \hat{f}$$

Let $\tilde{f}(\underline{\omega})$ denote the multidimensional Fourier transform of $f(\underline{x})$; $\underline{x} \in \mathbb{R}^{m_0}$; $\underline{\omega}$ is the frequency vector

$$f(\underline{x}) = \int_{\mathbb{R}^{m_0}} \tilde{f}(\underline{\omega})\, e^{j \underline{\omega}^T \underline{x}}\, d\underline{\omega}$$

For a complex valued fn $\tilde{f}(\underline{\omega})$ for which $\underline{\omega}\, \tilde{f}(\underline{\omega})$ is integrable, let

$$C_f = \int_{\mathbb{R}^{m_0}} \left| \tilde{f}(\underline{\omega}) \right| \, \left\| \underline{\omega} \right\|^{\frac{1}{2}} \, d\underline{\omega}$$

First absolute moment of the Fourier mag. distribution of $f$

$C_f$ measures Smoothness of $f$

Consider a ball $B_r = \{ \underline{x} : \|\underline{x}\| \le r \}$ $r > 0$

Theorem : For every cont. function $f(\underline{x})$ with finite first moment $C_f$ and every $m_1 \geqslant 1$ $\exists$ a linear combination of sigmoid based functions $F(\underline{x})$ / when $f(\underline{x})$ is observed over i/ps $\underline{x}$ $\{\underline{x}_i\}_{i=1}^{N}$ restricted to $B_r$, the empirical risk

$$\mathcal{E}_{av} = \frac{1}{N} \sum_{i=1}^{N} \left( f(\underline{x}_i) - F(\underline{x}_i) \right)^2 \le \frac{(2 r C_f)^2}{m_1}$$

size of hidden neurons

$$\mathcal{E}_{av}(N) \le O\left(\frac{C_f^{\,2}}{m_1}\right) + O\left(\frac{m_0\, m_1}{N}\log N\right)$$

# examples

$$m_1 \simeq C_f\left(\frac{N}{m_0 \log N}\right)^{\frac{1}{2}}$$

$$\mathcal{E}_{av}(N) = O\left(\left(\frac{1}{N}\right)^{\frac{2s}{2s+m_0}}\right)$$

s is a measure of smoothness

$$O\left(\left(\frac{1}{N}\right)^{\frac{1}{m_0}}\right)$$

$$O\left(\frac{1}{N^{1/m_0}}\right)$$ density

Sampling density $\propto N^{\frac{1}{m_0}}$

$\Rightarrow$ We need to densely sample the data points to learn the function well.

$\Rightarrow$ Higher dim $\Rightarrow$ Exponential growth in the complexity of the n/w algo.

# A few other technicalities

1) Jacobians
2) Hessians

**Jacobian** : Let $W$ denote the total # of free parameters i.e., the synaptic weights and biases of a MLP to form a vector $\underline{W}$.

Let $N$ denote the examples presented to the n/w.

Using the BPA, we can get $F(\underline{W}, \{\underline{x}\})$

Suppose $\{\underline{x}_n\}_{n=1}^{N}$

$$J := \left[ \frac{\partial F(\underline{w}, \underline{x}_n)}{\partial w_{ij}} \right] \qquad \begin{aligned} i &= 1, \dots, W \\ j &= 1, \dots, N \end{aligned}$$

<u>Empirically</u> rank $(J)$ can decide the efficiency of the BPA

If $J$ is not full rank (rank deficiency)

$\Longrightarrow$ longer training times

## Hessians

$$H := \left[ \frac{\partial^2 \mathcal{E}_{av}(\underline{w})}{\partial \underline{w}^2} \right]$$

Why do we need to study Hessians?

1) Eigen values of Hessian have a role in the BPA dynamics.

- Inverse of H can provide a basis for pruning/deleting insignificant wts.

2) Lead to 2nd order opt. methods.

The Hessian of an error surface has :

(Empirical)

(a) Small # of small & large sized eigen values

(b) large # of med. sized eigen values

whose composition depend

(1) non zero mean of i/p signals & induced neural o/p signals
(non zero)

(2) Correlations between various attributes of a data vector

(3) Wide variations in the 2nd order derivatives of the $J$ w.r.t $\underline{W}$ from one layer to the other.

# Complexity regularization

In the context of the back prop. algo.; we may want to minimize the foll. risk

$$R(\underline{w}) = \underbrace{\mathcal{E}_{av}(\underline{w})}_{\text{standard performance metric}} + \underbrace{\lambda}_{\text{scalar}} \underbrace{\mathcal{E}_c(\underline{w})}_{\text{complexity part}}$$

$\mathcal{E}_c(\underline{w})$ is the complexity penalty measured in terms of $\underline{w}$

$$\mathcal{E}_c(\underline{w}) = \|\underline{w}\|^2$$

(Choice of min $\mathcal{E}_c(\underline{w})$, force some wts. to zero & permit larger weights)

## Alternative Strategies to learning

$$\mathcal{E}_{av}(\underline{w}+\Delta\underline{w}) = \mathcal{E}_{av}(\underline{w}) + g^T(\underline{w})\,\Delta\underline{w} + \frac{1}{2!}\Delta\underline{w}^T \overset{\text{2nd order}}{H}\, \Delta\underline{w} + h.o.t.$$

a small perturbation $\Delta\underline{w}$ around $\underline{w}$.

We can set

$$\frac{\partial \mathcal{E}_{av}(\underline{w}+\Delta\underline{w})}{\partial \Delta w} = 0 \implies \boxed{\Delta\underline{w} = -H^{-1}\underline{g}}$$

# Optimum brain Surgeon Algo

$$\min_{\Delta \underline{w}} \frac{1}{2} \Delta \underline{w}^T \mathbb{H} \Delta \underline{w}$$

1) We set the ith comp. of $\Delta \underline{w}$ to zero & min. over all synaptic wt. vectors that remain

2) Do this min. again over all indices 'i'

# Cross validation strategies

Need: We may get a low training error over a data set, but the error may be pronounced over a different data set. To get a reasonable performance, we divide the training sets into groups, train & test over the groups. "Empirical soln" in the absence of a "theoretical" set up

Eg: 1) Leave 1 out strategy: Train on (N - 1) samples & test on the others. Do over $\underbrace{\text{all} \binom{N}{1} \text{choices}}$

2) K-fold cross validation: Divide N samples into K equal groups, Train on (K-1) groups & test on the other, Repeat over all the choices

# Convolution networks

**Motivation :** Structural layout of the MLP algo. with preprocessing steps that are neurobiologically inspired.

Ex: Work from Hubel & Wiesel related to specialized neurons in the visual cortex of a cat.

Simple cells — locally sensitive

Complex cells — Orientation

etc

GOAL : To design a MLP to recognize 2D/3D shapes/objects with high invariance to translation/rotation/scaling.

1) Feature extraction : Each neuron takes its inputs from local receptive fields in previous layers, forcing to extract local features.

Once the features are extracted, its exact location is less important as long the relative position w.r.t. other features is preserved.

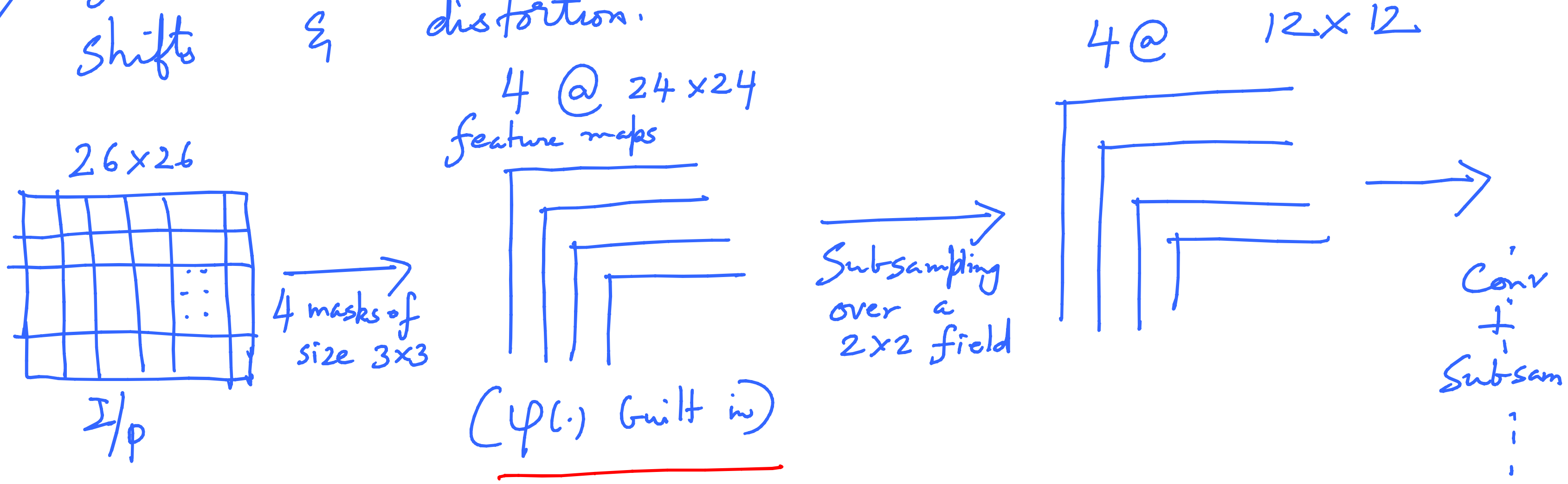
2) Feature mapping: Each computational layer is composed of multiple feature maps, with each feature map being in the form of an appropriate geometry to the signal (e.g., plane for images etc.) & constrained to share the same synaptic weights

a) Shift Invariance: Forced into the feature map through convolution with a kernel of a small size

b) Reduction in the # of free parameters. This is ensured via wt. sharing.

3) Sub sampling: Each conv. layer performs some local operations followed by a non-linear activation $\varphi(.)$ & then sub sampling that reduces the resolution of the feature maps to

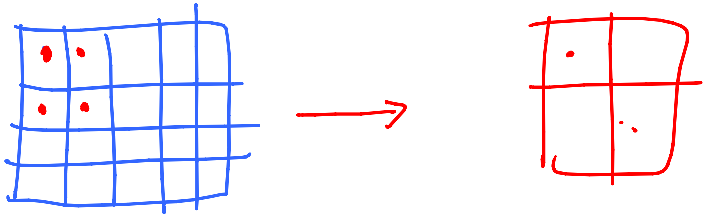
⇒ Tolerance to sensitivity shifts & distortion.



Pyramidal Effect !

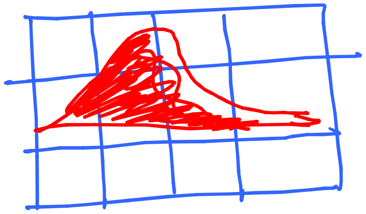
Conv. followed by subsampling is inspired by simple cells followed by complex cells as described by Hubel & Wiesel.

As # of subsampling layers increase ↑ spatial resolution ↓ compared to what you had in the prev. layer



## Let us understand the Computations through an example

1) Image :



3x4

→ I/p to an array of integers

|   |   |   |   |
|---|---|---|---|
| 0 | 1 | 1 | 0 |
| 1 | 1 | 1 | 1 |
| 0 | 0 | 0 | 0 |

2) Filter the image by masks / feature masks to get the features ( low pass to high pass )

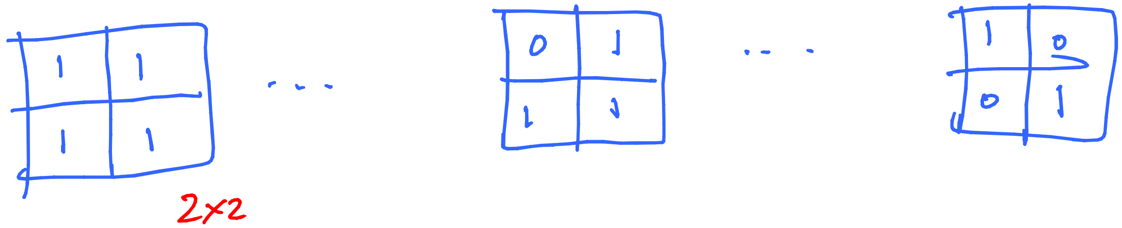
More Low pass

(a)

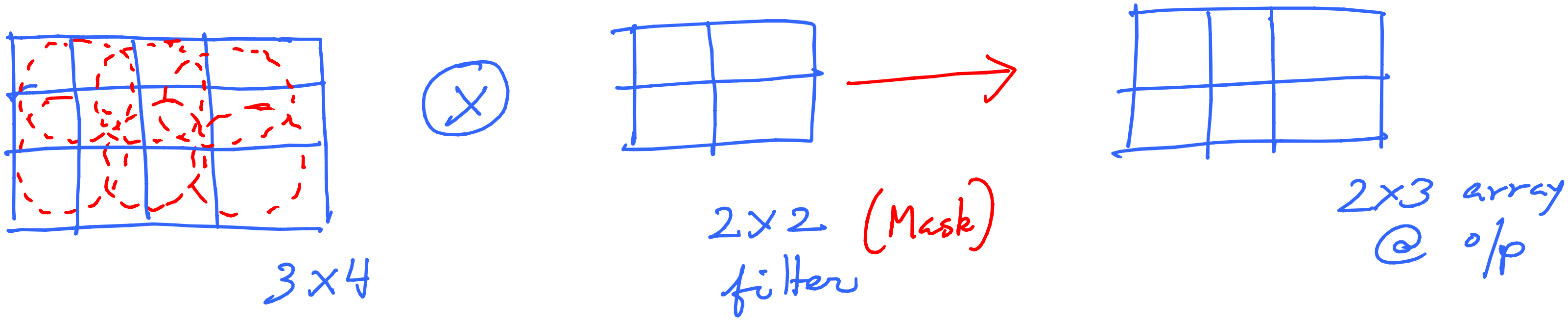
(b)

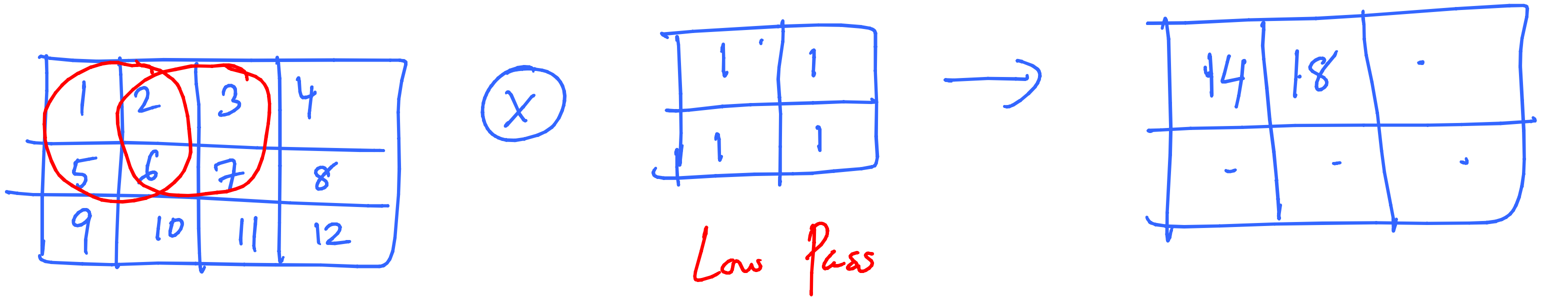
High Pass

## Types of masks :

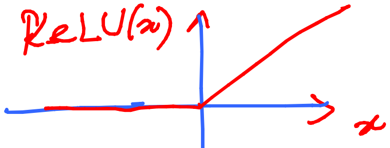


Eg:

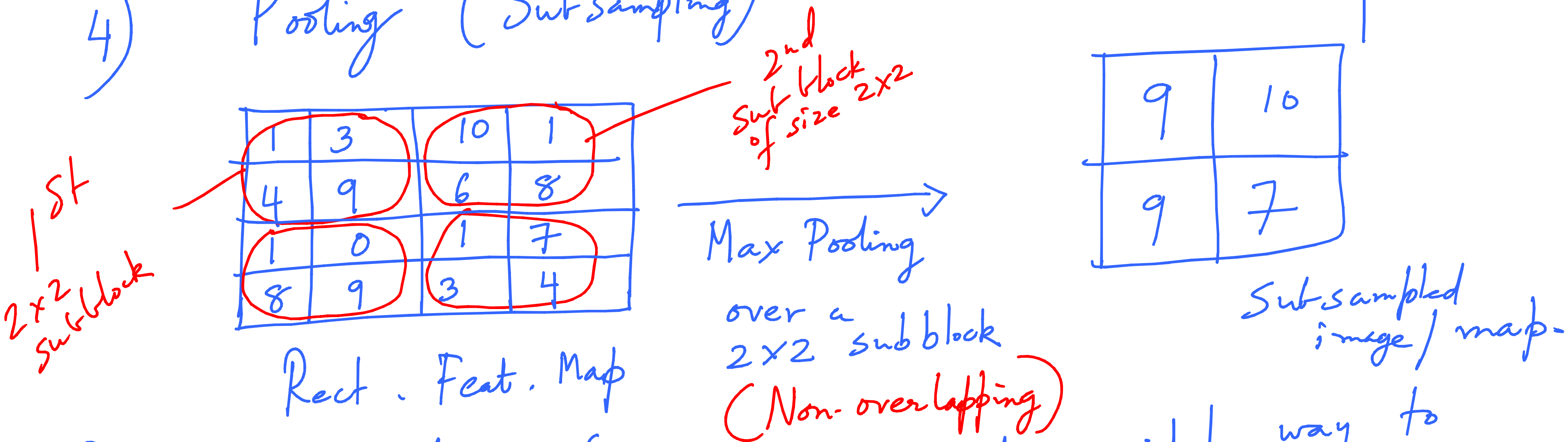




3) Once the features are extracted, we feed this to a non-linear act. fn ReLU $(x) = \max(0, x)$



4) Pooling (Subsampling)

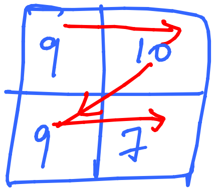


| | | | |
|---|---|---|---|
| 1 | 3 | 10 | 1 |
| 4 | 9 | 6 | 8 |
| 1 | 0 | 1 | 7 |
| 8 | 9 | 3 | 4 |

| | |
|---|---|
| 9 | 10 |
| 9 | 7 |

5) Iterate steps (2) - (4) in a pyramidal way to get the final size as desired.

Rastering



$$\longrightarrow \begin{bmatrix} 9 \\ 10 \\ 9 \\ 7 \end{bmatrix} \in \mathbb{R}^4$$

a feature vector

↓ fed into the MLP

desired response →

## Masks can be adaptive

|  |  |
|---|---|
| $w_{1,1}$ | $w_{1,2}$ |
| $w_{1,3}$ | $w_{1,4}$ |

Conditions on the elements within the mask

⟹ Regularized n/w for learning I/o mappings

# Cover's Theorem

Ref 1 : Thomas Cover, Feb 1964

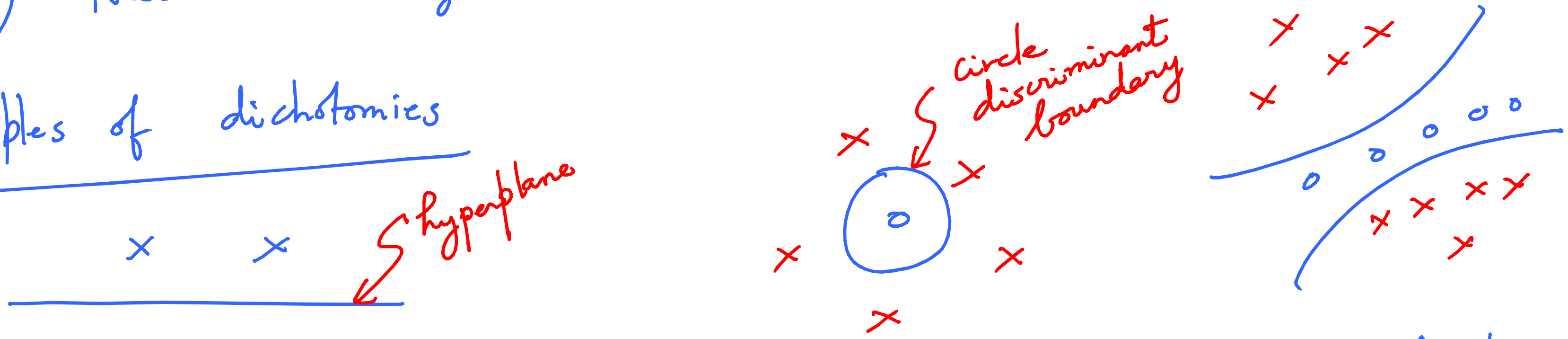
Ref 2 : Nils Nilsson, Learning Machines, 1965

## Motivation :

a) How do we quantify the complexity of a neural network architecture?

b) Need a counting measure for a discrete set of mappings.

## Examples of dichotomies



(a) Linear dichotomy

(b) Spherical dichotomy

(c) Quadratic dichotomy

Consider a fixed finite set of input vectors $\{\underline{x}_1, \ldots, \underline{x}_p\}$; $\underbrace{\underline{x}_i \in \mathbb{R}^N}$

Can we attempt to compute the dichotomies for a perceptron?

The # of linearly realizable dichotomies on the set of points depends on a mild condition called `general position'

General position demands that no sub-set of size $\leq N$ on $\{\underline{x}_1, \underline{x}_2, \ldots, \underline{x}_p\}$ is linearly dependent

Theorem : Let $\{\underline{x}_1, \ldots \underline{x}_P\}$ be vectors in $\mathbb{R}^N$ that are in a general position. The # of distinct dichotomies applied to these points can be realized by a hyperplane is

$$\underbrace{C(P, N) = 2\sum_{k=0}^{N} \binom{P-1}{k}}$$

Proof: We start with $P$ points in general position. Let us assume that there are $C(P, N)$ dichotomies on them.

↑ counting fn.

Suppose we add an extra point to this set.

We need $C(P+1, N)$ dichotomies.

Idea: Set up a recursion to link $C(P+1, N)$ with $C(P, N)$.

Let $(\sigma_1, \sigma_2, \dots, \sigma_p)$ be a dichotomy realizable by a hyperplane over the set of $P$ inputs. $\sigma_i \in \{-1, 1\}$ for every $i = 1, \dots, P$. There is a set of weights $\underline{w}$ so that

$$\Big(\text{sign}(\underline{w}^T \underline{x}_1), \text{sign}(\underline{w}^T \underline{x}_2), \ldots, \text{sign}(\underline{w}^T \underline{x}_p) = \underbrace{(b_1, b_2, \ldots, b_p)}_{}$$

Using one such $\underline{w}$, we get a dichotomy over $P+1$ points

i.e., $\Big(\underset{\uparrow\, b_1}{\text{sign}(\underline{w}^T \underline{x}_1)}, \ldots, \underset{\uparrow\, b_p}{\text{sign}(\underline{w}^T \underline{x}_p)}, \text{sign}(\underline{w}^T \underline{x}_{p+1})\Big)$

For every linearly realized dichotomy over $P$ points, there is at least one linearly realized dichotomy over $P+1$ points.

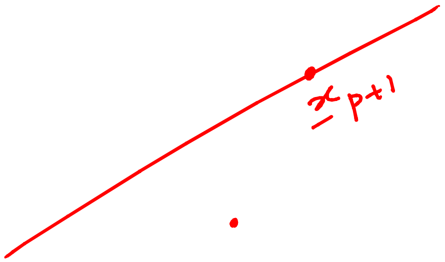
Different dichotomies over $P$ points define different dichotomies over $P+1$ points since they differ some where over first $P$ coordinates

Note that the additional dichotomy $(b_1, \dots, b_p, \underset{\sim}{-}\,\text{sign}(\underline{w}^T \underline{x}_{p+1}))$ is also possible by reversing the sign of the last coordinate.

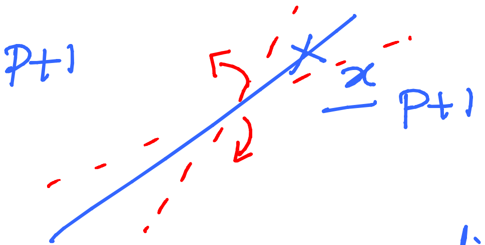
$$\Longrightarrow \quad C(P+1, N) > C(P, N)$$

Let $C(P+1, N) = C(P, N) + E$ ← extra dichotomies possible.

There are $\underline{2\ \text{cases}}$ to consider



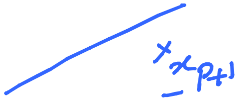
Case A: One of the weight vectors $\underline{w}$ that generates $(b_1, b_2, \ldots, b_p)$ passes through $\underline{x}_{p+1}$



By adjusting the angle of the hyperplane, we can adjust $\text{sign}(\underline{w}^T \underline{x}_{p+1})$ to $+1$ or $-1$

i.e., both $(b_1 \ldots b_p\ +1)$ and $(b_1 \ldots b_p\ -1)$ are also possible!

Case B: No hyperplane passes through $\underline{x}_{p+1}$ and generates $b_1 \ldots b_p$ on first $p$ vectors.



⇒ Point lies on one side of the old dichotomy

E is the # of dichotomies over $p$ points that are realized by a hyperplane passing through a fixed point $\underline{x}_{p+1}$. By forcing the hyperplane to pass through $\underline{x}_{p+1}$, we are going to $N-1$ dimensions instead of $N$

Geometrically, if a point is on the $x$-axis, the hyperplane has $N-1$ axes left to work on the problem. If it is not on the $x$-axis, then rotate the axes of the space to get the point on the $x$-axis and there is no effect on the geometry of the problem

$$\therefore \quad E = C(P, N-1)$$

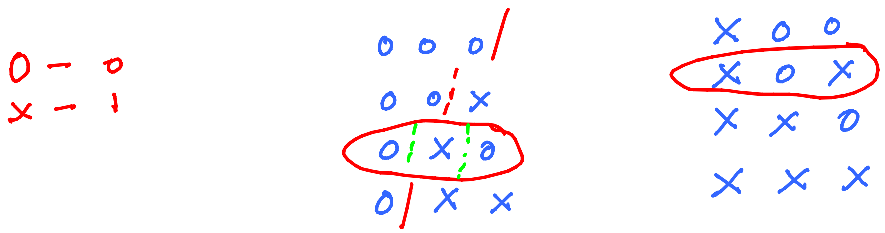
$$\therefore \quad C(P+1, N) = C(P, N) + \underbrace{\overset{E}{C(P, N-1)}} \qquad \text{———} \quad (1)$$

Let us consider the boundary conditions

$C(1, N) = 2$ ✓ (There is 1 point in $\mathbb{R}^N$ & can be realized by 2 labels)

$C(P, 1) = 2P$ ✓ (There are $P$ points in $\mathbb{R}^1$)

Consider $P = 3$, we have



Observe that except oxo and xox, we can have a hyperplane that can shatter the points!

Let us prove the result through induction.

(We are doing induction over 'P')

Base case :

Case 1 : $C(1, N) = 2$ as expected

There is 1 point in N dim. Follows from one of the boundary conditions

Induction : $$C(P+1, N) = \underbrace{2\sum_{k=0}^{N}\binom{P-1}{k}}_{C(P,N)} + \underbrace{2\sum_{k=0}^{N-1}\binom{P-1}{k}}_{C(P,N-1)}$$

Ponder on $\binom{P-1}{0}$ case

Meaning of '0' dim.

Given by statement of Cover's theorem

Now, simplifying,

$$= \underbrace{2\sum_{k=0}^{N}\binom{p-1}{k}}_{} + \underbrace{2\sum_{k=0}^{N}\binom{p-1}{k-1}}_{} \qquad \left(\because \binom{p-1}{-1} = 0\right)$$

$$= 2\left[\sum_{k=0}^{N}\left[\binom{p-1}{k} + \binom{p-1}{k-1}\right]\right]$$

$$= \underline{\underline{2\sum_{k=0}^{N}\binom{p}{k}}} \qquad \left(\because \binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}\right.$$

Basic identity from elementary Combinatorics$\left.\right)$ ▨

## Implications :

Let us consider the prob. of having a perceptron that (linear decision boundary) can provide a linear dichotomy over $P$ points

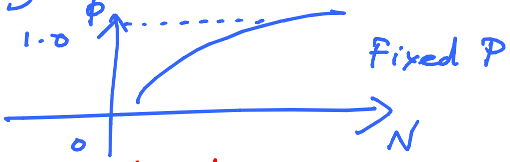
$$\phi = \frac{C(P, N)}{2^P}$$

$2^P$ ← total # of dichotomies (not necessarily linear)

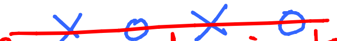
$$= 2^{1-P} \sum_{k=0}^{N} \binom{P-1}{k}$$

Home Work : Plot $\phi$ vs. dim $N$ for a fixed 'P'

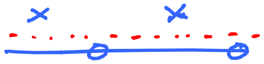
Observe the Concavity of $\phi$



$\mathbb{R}^1$ –



(Linear decision boundary is not possible)



$\mathbb{R}^2$ – ✓ Yes, a linear decision boundary is possible

# Let us revisit the XOR problem

**Recall:** Cover's theorem has 2 important consequences

1) The use of a non-linear function i.e., a hidden function defined by $\varphi(\underline{x})$ ← acts on the input vector

2) High dimensionality of the hidden/feature space compared to the input space

$$\left.\begin{array}{l} N := \text{dim. feature space} \\ M := \text{dim. input space} \end{array}\right\} \qquad N > M$$

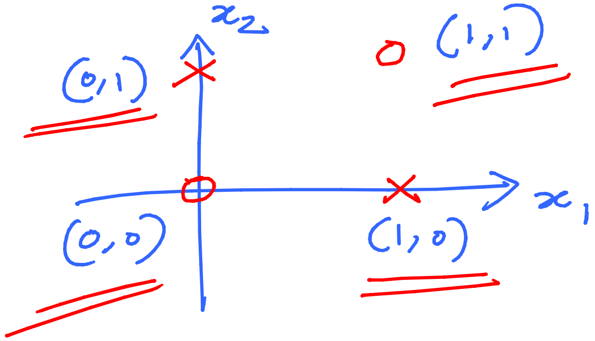
Recall: A dichotomy is $\varphi$- separable if $\exists$ a N-dim. vector $\underline{w}$ /

non-linear fn.

$\underline{w}^T \phi(\underline{x}) > 0$ $\quad \underline{x} \in$ Class 1

$\underline{w}^T \phi(\underline{x}) \leq 0$ $\quad \underline{x} \in$ Class 2

We have the XOR problem



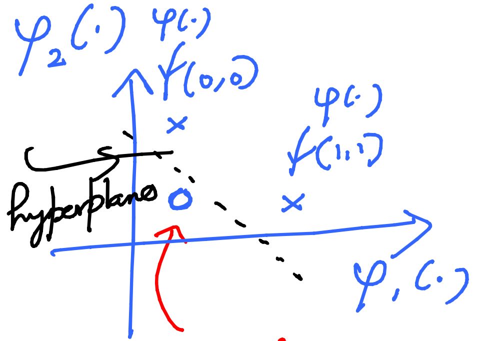
Let us consider the pair of Gaussian hidden functions.

$$\varphi_1(\underline{x}) = \exp\left(-\|\underline{x}-\underline{t}_1\|^2\right); \quad \underline{t}_1 = [1, 1]^T$$

$$\varphi_2(\underline{x}) = \exp\left(-\|\underline{x}-\underline{t}_2\|^2\right); \quad \underline{t}_2 = [0, 0]^T$$

Let us tabulate the comp. evaluations of $\underline{x}$ over $\varphi_1$ & $\varphi_2$

| $\underline{x}$ | $\varphi_1(\underline{x})$ | $\varphi_2(\underline{x})$ |
|---|---|---|
| (1, 1) | 1 | 0 . 1353 |
| (0, 1) | 0.3678 | 0.3678 |
| (0, 0) | 0.1353 | 1 |
| (1, 0) | 0.3678 | 0.3678 |



Both points (0,1) & (1,0) map to the same point.