PERCEPTRON LEARNING ALGORITHM →

LEARN $(\vec{x}^\mu, y^\mu)$, $y \in \{-1,1\}$ BY FINDING w ST. $y^\mu = \Theta(\vec{w}\cdot\vec{x}^\mu)$

WORKS ON LINEARLY SEPARABLE DATA

USE $\begin{cases} w_0 = -T, \\ x_0^\mu = 1 \end{cases}$ $\Big|\ \Theta(\cdot) = I(x \geq 0)$

(ASSUMING SOLUTION EXISTS)

1. INITIALISE $\vec{w}_0 = 0$, η (LEARNING RATE)
2. REPEAT UNTIL CORRECT:
   - A. PICK A DATAPOINT μ'
   - B. IF $\vec{w}_t$ MISCLASSIFIES $y^{\mu'}$

     $\vec{w}_{t+1} = \vec{w}_t + y^\mu \vec{x}^\mu$ (ALIGNS THE PLANE'S NORM TO THE POINT)

PROOF OF CONVERGENCE:

↓ FIND UPPER AND LOWER BOUNDS FOR w^{k+1} AND SOLVE FOR A BOUND ON k

TERMINATES IF ⊕ POINTS ARE SEPARATED FROM ⊖

ITERATIONS TO CONVERGE $k \leq R^2/\gamma^2$ ← $\gamma = \inf_i \left[y^i (w^* \cdot x^i) \right]$ ← LEAST ALIGNMENT BETWEEN DATA AND POINTS

LOWER BOUND

$w_1 = 0$, IF $(k \geq 1)$ x^j IS MISCLASSIFIED $\quad w^{k+1}\cdot w^* = (w^k + y^j x^j)\cdot w^*$

$\overset{!}{=} w^k \cdot w^* + y^j (x^j \cdot w^*)$

USING $w^{k+1}\cdot w^* \leq \|w^{k+1}\|\cdot\|w^*\|$ $\qquad \overset{!}{>} w^k \cdot w^* + \gamma \Rightarrow w^{k+1}\cdot w^* > k\gamma$

(INDUCTION) $\|w^{k+1}\| \geq k\gamma$

UPPER BOUND

y MUST BE MISCLASSIFIED

$\|w^{k+1}\|^2 = \|w^k + y^j x^j\|^2 = \|w^k\|^2 + \|x^j\|^2 + 2(w^k \cdot x^j) y^j \leq \|w^k\|^2 + \|x^j\|^2$

(INDUCTION) $\|w^{k+1}\|^2 \leq kR^2$ $\qquad \overset{!}{\leq} \|w^k\|^2 + R^2$

COMBINATION $\boxed{k^2\gamma^2 \leq \|w^{k+1}\|^2 \leq kR^2 \rightarrow k < \frac{R^2}{\gamma^2}}$

SVM

PRIMAL FORMULATION

PERCEPTRON ALG. WITH WIDEST MARGIN (STABLE WITH ERRORS)

$\max_w \frac{2}{\|w\|}$ (MARGIN) ST. $y_n(w^T x_n + b) \geq 1$

$\min_{w,b,\xi} \frac{1}{2}\|w\|^2 + C\sum_{n=1}^N \xi_n$ ← (SOFT) MARGIN

ST. $y_n(w\cdot x_n + b) \geq 1 - \xi_n$

(SMALL C, SOFT MARGIN) $\xi_n \geq 0$

LEARNING: USE HINGE LOSS $\xi \mapsto \max\{0, \xi_n\}$

CONVEX PROBLEM OBJECTIVE IS SUM OF TWO CONVEX FUNCTIONS

→ DERIVATION OF GRADIENT DESCENT: $w_{t+1} = w_t - \eta_t \nabla_w C(w_t)$ $\quad \left(\text{USING } \lambda = \frac{1}{NC}\right)$

$C(w) = \frac{1}{N}\sum_{n=1}^N \left(\frac{\lambda}{2}\|w\|^2 + \max(0, 1 - y_n(w\cdot x_n) + b)\right)$

DUAL EXPLANATION: NUMBER OF PARAMETERS INCREASES WITH THE NUMBER OF SAMPLES → USE THE DUAL IF YOU HAVE FEW DATA-POINTS, WITH MANY FEATURES

$\min_\alpha \frac{1}{2}\sum_{i=1}^N\sum_{j=1}^N y_i y_j \alpha_i \alpha_j x_i \cdot x_j - \sum_{i=1}^N \alpha_i$

ST. $\sum_{i=1}^N y_i \alpha_i = 0 \quad \forall i \quad 0 \leq \alpha_i \leq C$

REPRESENTER THEOREM DERIVATION:

WRITE LAGRANGIAN OF SOFT MARGIN $(a_n, \xi_n \geq 0)$

$\mathcal{L}(w,b,\xi,\alpha,\gamma) = \frac{1}{2}\|w\|^2 + C\sum_{n=1}^N \xi_n - \sum_{n=1}^n a_n\left(y_n(w\cdot x_n + b) - 1 + \xi_n\right) - \sum_{n=1}^N \gamma_n \xi_n$

$\frac{\partial \mathcal{L}}{\partial w} = w^T - \sum_{n=1}^N a_n y_n x_n^T = 0$

$w = \sum_{n=1}^N a_n y_n x_n$

OPTIMAL VECTOR IS IN THE SPAN OF THE DATA

- POLY DEG $\leq d$: $(x\cdot y + 1)^d$
- RBF $= \exp\left(-\frac{\|x-y\|^2}{2\sigma^2}\right)$

NON-LINEAR/KERNEL TRICK → NON-LINEARLY SEPARABLE DATA CAN FIRST BE MAPPED TO MAKE IT SEPARABLE

KERNEL IS MAINTAINED BY PRODUCT, CONIC COMBINATION

KERNEL FUNCTION: PRODUCES THE DOTTED RESULT WITHOUT THE INTERMEDIATE STEPS OF COMPUTATION

$k(x,y) = \phi(x)\cdot\phi(y)$

- SYMMETRY $(k(x,y) = k(y,x))$
- POS-DEF: $K_{ij} = k(x^{(i)}, x^{(j)})$ ← REPRESENTS WHAT DATA THE ALGORITHM SEES

  IS POSITIVE SEMIDEF ⇓ ENSURES CONVEXITY

RNN

J_{ij} SYMMETRIC CONNECTION MATRIX, $J_{ii}=0$

SYMMETRIC $\Rightarrow$ ENERGY F.
⊕ PARALLEL IN RANDOM TIME TO LOWER E.

ENERGY FUNCTION (SYMMETRIC NETWORKS): $E(S_1 \dots S_N) = -\frac{1}{2}\sum_{j \neq i} J_{ij} S_i S_j$ → DECREASES MONOTONICALLY TO A MINIMUM

$$S_k(t+dt) = \mathrm{SIGN}\left(\sum_j J_{kj} S_j(t)\right)$$

ASYNC. DYNAMIC → PICK k RANDOM NEURON TO UPDATE

J_{kj} ← WEIGHTS ON THE EDGES

HOPFIELD: N NEURONS (±1)
P PATTERNS TO MEMORISE: $\xi_i^\mu = \pm 1$ (IN THE WHOLE SYSTEM) → μ : PATTERN N° } A.K.A. MEMORY; i : VERTEX N°

HEBBIAN (≈ CORRELATION) MATRIX: $J_{ij} = \frac{1}{N}\sum_\mu \xi_i^\mu \xi_j^\mu$ (CALCULATED OVER ALL THE MEMORIES)

→ CONDITIONS FOR FIXED POINT ATTRACTORS: A CORRELATED STATE CONVERGES TO THAT COMBINATION OF STATES | CONVERGES TO PATTERN IN ONE UPDATE IF THERE IS A FINITE OVERLAP → PATTERN COMPLETION

→ SIGNAL TO NOISE ANALYSIS: $S_i(1) = \mathrm{SIGN}\left(\underbrace{\xi_i^1}_{\text{MAIN SIGNAL}} + \underbrace{\frac{1}{N}\sum_{j,\mu>1} \xi_i^\mu \xi_j^\mu \xi_j^1}_{\text{NOISE}}\right)$ } NOISE IS CAUSED BY OTHER PATTERNS INTERFERING

SYSTEM IS INITIALISED IN A STORED PATTERN

t TIME

MEAN/VARIANCE → $\left(E=0,\ V=\frac{p-1}{N}\right)$ ← FOR NUMBER OF MEMORIES

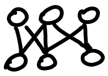
→ CAPACITY IS LINEAR WITH # NEURONS AND LIMITED BY STATES (LOCAL MIN) MERGING AND CREATING SPURIOUS MEMORIES
↓
NEARBY/CORRELATED PATTERNS MERGE INTO A NEW MINIMUM

→ NOISE ALLOWS TO CROSS BARRIERS
⇓
SIMULATED ANNEALING PRIORITISES TRANSITION TO LOWER PROBABILITIES

$P = 0.14 \cdot N$ CAPACITY ⇒ OTHERWISE THERE IS TOO MUCH INTERFERENCE

RBM } → ENCODING/DECODING } KL-DIVERGENCE | SYSTEM REACHES THERMAL EQUILIBRIUM IN ONE STEP

→ DATA VECTORS ARE BINARY



(BIPARTITE GRAPH)

1. LEARNING: SHOWN DATA VECTORS THAT IT MUST GENERATE WITH HIGH $\mathbb{P}$/ACCURACY
2. GENERATION: FIX CONNECTION WEIGHTS, USE ENERGY, SAMPLE LOW E STATES

COST FUNCTION:

$z_i = b_i + \sum_j w_{ij} S_j$ (STATE UPDATE)

$\mathbb{P}(S_i = 1) = \frac{1}{1+e^{-z_i}}$ (STATE PROBABILITY)

$P(S) = \frac{1}{Z}\exp(-E(S))$ (RANDOM UPDATING → BOLTZMANN DISTRIBUTION)

$Z = \sum_S \exp(-E(S))$ (PARTITION FUNCTION)

$E(S) = -\sum_{i<j} w_{ij} S_i S_j - \sum_i b_i S_i$

GRADIENT DESCENT: $\max C = -\frac{1}{M}\sum_\mu \log P(x^\mu)$ (LOG-LIKELIHOOD OVER THE DATA) ↳ BOLTZMANN DISTRIB.

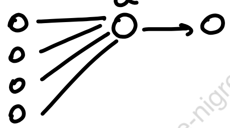
DERIVATION OF LEARNING ALGORITHM: $\frac{\partial C}{\partial \vartheta} = \frac{1}{M}\sum_\mu \mathbb{E}_h\left(\frac{\partial E(x^\mu, h)}{\partial \vartheta}\right) - \mathbb{E}_{x,h}\left(\frac{\partial E(x,h)}{\partial \vartheta}\right)$

↳ MONTE CARLO SIMULATION/METHODS ARE USED TO APPROXIMATE THE $\mathbb{E}(\dots)$

CONTRASTIVE DIVERGENCE: REPLACE AVERAGE OF NEGATIVE TERM WITH AN ESTIMATE (GIBBS SAMPLING) IN k STEPS
⊕
BATCHES OF SIZE k ARE USED TO COMPUTE EACH ESTIMATE

AUTOENCODERS
- COST IN DIMENSIONALITY REDUCTION → $\ell(x, f(x))$ RECONSTRUCTED
- DENOISING → TRAIN ON $x+\varepsilon$, $\varepsilon \sim N(\underline{0}, \sigma^2 I)$

MULTILINEAR NN → ARCHITECTURE LIMIT



$$z_j^L = \sigma(a_j^{(L)} + b) = \sigma(a^{(L)} + w_{j0}^{(L)}) = \sigma\left(\sum_{i=1}^{d^{(L-1)}} w_{ji} z^{(L)} + w_{j0}^{(L)}\right)$$

SURROGATE LOSS → (NP-HARD) 0-1 LOSS ⇓ USE A RELAXATION } ℓ_2 LOSS FOR CONVEXITY

ACTIVATION FUNCTIONS

RELU $= \max(0, z)$ LOGISTIC $= \frac{1}{1+e^{-z}}$

SOFT PLUS $= \log(1+e^{z})$, HYPERB. TANG $= \frac{e^{z}-e^{-z}}{e^{z}+e^{z}}$

HARD THRESH $= I(z>0)$

KORNIK THM STATEMENT → $\sigma : \mathbb{R} \to \mathbb{R}$ NONCONST BOUNDED NONDECREASING CONTINUOUS }4

$S \subseteq \mathbb{R}^d$ CLOSED BOUNDED }2

$f: S \to \mathbb{R}$, $\forall \varepsilon > 0$

$\exists$ NN WITH ONE HIDDEN LAYER, FINITE HIDDEN LAYERS

$$h(\underline{x}) = \sum_{j=1}^{k} w_j^{out} \sigma(w_j^T x + b_j) \quad \text{ST.} \quad |h(\underline{x}) - f(x)| < \varepsilon \quad \forall x \in S$$

- PROOF EXISTENCE
- NO IDEA ON ARCHITECTURE

⇓

MORE LAYERS GIVES MORE EXPRESSIVE FOR LESS COST PER NEURON

XOR EXAMPLE

TRAIN NN TO CLASSIFY XOR : $\phi\begin{pmatrix} x_1 \\ x_2 \\ x_1 x_2 \end{pmatrix}$ →

X	XOR	$\phi(x)$
0 0	0	0 0 0
0 1	1	0 1 0
1 0	1	1 0 0
1 1	0	1 1 1

→ USE: $\underline{w} = \begin{pmatrix} -0.2 \\ 0.5 \\ 0.5 \\ -1.5 \end{pmatrix}$, $x = \begin{pmatrix} 1 \\ x_1 \\ x_2 \\ x_3 \end{pmatrix}$

BACKPROPAGATION ALGORITHM GENERAL SCHEME

FIRST LAYER FEATURE LEARNING → FILTERS EXTRACT BASIC FEATURES AT THE FIRST LAYERS

EFFICIENT SCHEME (DERIVATION)
- FORWARD PASS : COMPUTE AND STORE WEIGHTS $x \mapsto f(x)$
- BACKWARD PASS : COMPUTE DERIVATIVES BACKWARDS

$\delta_j^{(\ell)} = \frac{\partial \ell(y, \hat{y})}{\partial a_j^{(\ell)}}$ BEST ESTIMATE

1. INITIALISE $\boxed{\delta^{(L+1)} = -(y-\hat{y}) = -(y - a^{(L+1)})}$ } ERROR

2. COMPUTE THE DERIVATIVES

$$\frac{\partial \ell(y, \hat{y})}{\partial w_j^{(L+1)}} = -(y-\hat{y}) z_j^{(L)} \qquad \frac{\partial \ell(y,\hat{y})}{\partial a^{(L+1)}}$$

↳ COMES FROM $\nabla(\ell^2)$, EQUAL TO IT UP TO A NORMALISATION

3. COMPUTE

YOU COMPUTE THE Δ BACKWARDS, USING THE KNOWLEDGE FROM THE LAST LAYER TO GET ALL PREVIOUS ONES

$$\left\{ \delta_j^{(\ell)} = \sigma'(a_j^{(\ell)}) \sum_{k=1}^{d^{(\ell+1)}} w_{kj}^{(\ell+1)} \delta_k^{(\ell+1)} \right\}$$ ACCOUNTS FOR ALL INFLUENCE ON

$$\frac{\partial \ell(y,\hat{y})}{\partial w_{ji}^{(\ell)}} = \delta_j^{(\ell)} z_i^{(\ell-1)}$$

SGD (FULL BATCH, MINI BATCH)

- CONVEX: $w_0 = \vec{0}$
- NOT CONVEX: $w_0 \neq$ CONST (OTHERWISE UPDATES ARE TOO CORRELATED)

$\vec{w}_{t+1} = \vec{w}_t - \eta \nabla f(w)$

(GD) $\underline{w}^{(k)} = \underline{w}^{(k-1)} + \eta^{(k)} \mathbb{E}_{\mu}\left[(y_n - \underline{w}^{(k-1)} \cdot \underline{x}_n) x_n\right]$ (USING SQUARE LOSS)

(SGD) SAME AS GD BUT ON SMALLER BATCH OF DATA } → IN BATCH SIZE → LOWER VARIANCE VS MORE COMPUTE

HEURISTICS
- DECAYING STEP SIZE
- UNSTABLE FOR BIG η

MULTI-CLASS CLASSIFICATION (SOFT-MAX) → PRESERVES MODEL CONVEXITY } SMOOTH FUNCTION

GENERALISES BINARY, PROVIDES "SMOOTHING"

$$p(y=\ell \mid x, w^{(1)} \ldots w^{(k)}) = \frac{\exp(w^{(\ell)} \cdot x)}{\sum_{i=1}^{k} \exp(w^{(i)} x)}$$