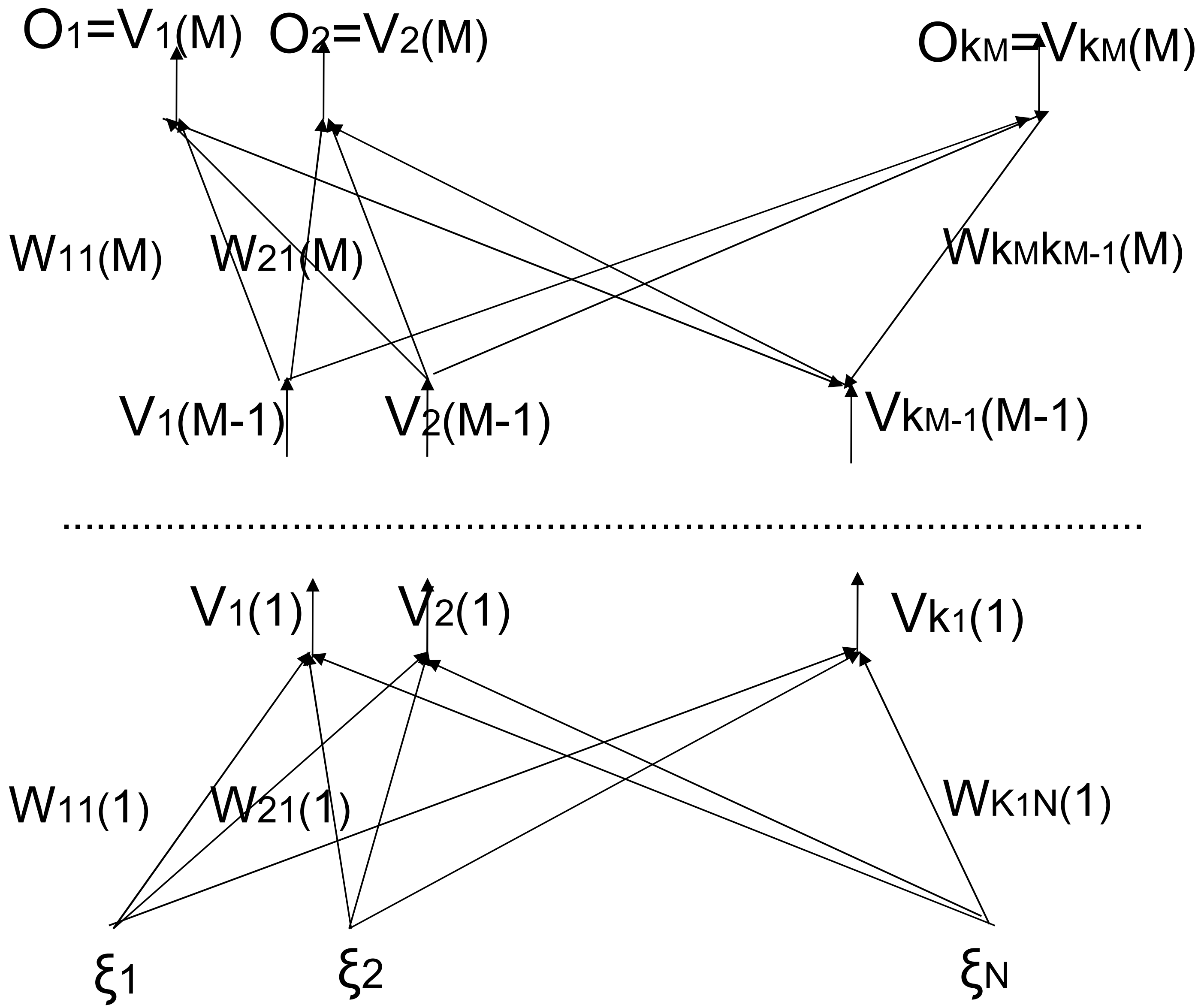
PERCEPTRON MULTICAPA



Si $M=1$ (perceptrón simple) $\begin{cases} \text{• fácil de entrenar} \\ \text{• soluciona familia de problemas restringida} \end{cases}$

Si $M \geq 2$ (perceptrón multicapa) $\begin{cases} \text{• mucho más difícil de entrenar (durante mucho tiempo no se supo cómo)} \\ \text{• puede representar cualquier función booleana y, más aun, continua en general} \end{cases}$

Teorema (Funahashi 1989):

Sean: g no constante, acotada y monótona creciente ; $C \subset \mathbb{R}^N$ compacto

$f : C \longrightarrow \mathbb{R}^L$

Dado $\varepsilon > 0$, existe K entero y constantes reales W_{ij}, θ_j $(j=1\ldots K, i=1\ldots L)$

w_{jk} $(j=1\ldots K, k=1\ldots N)$ tales que si

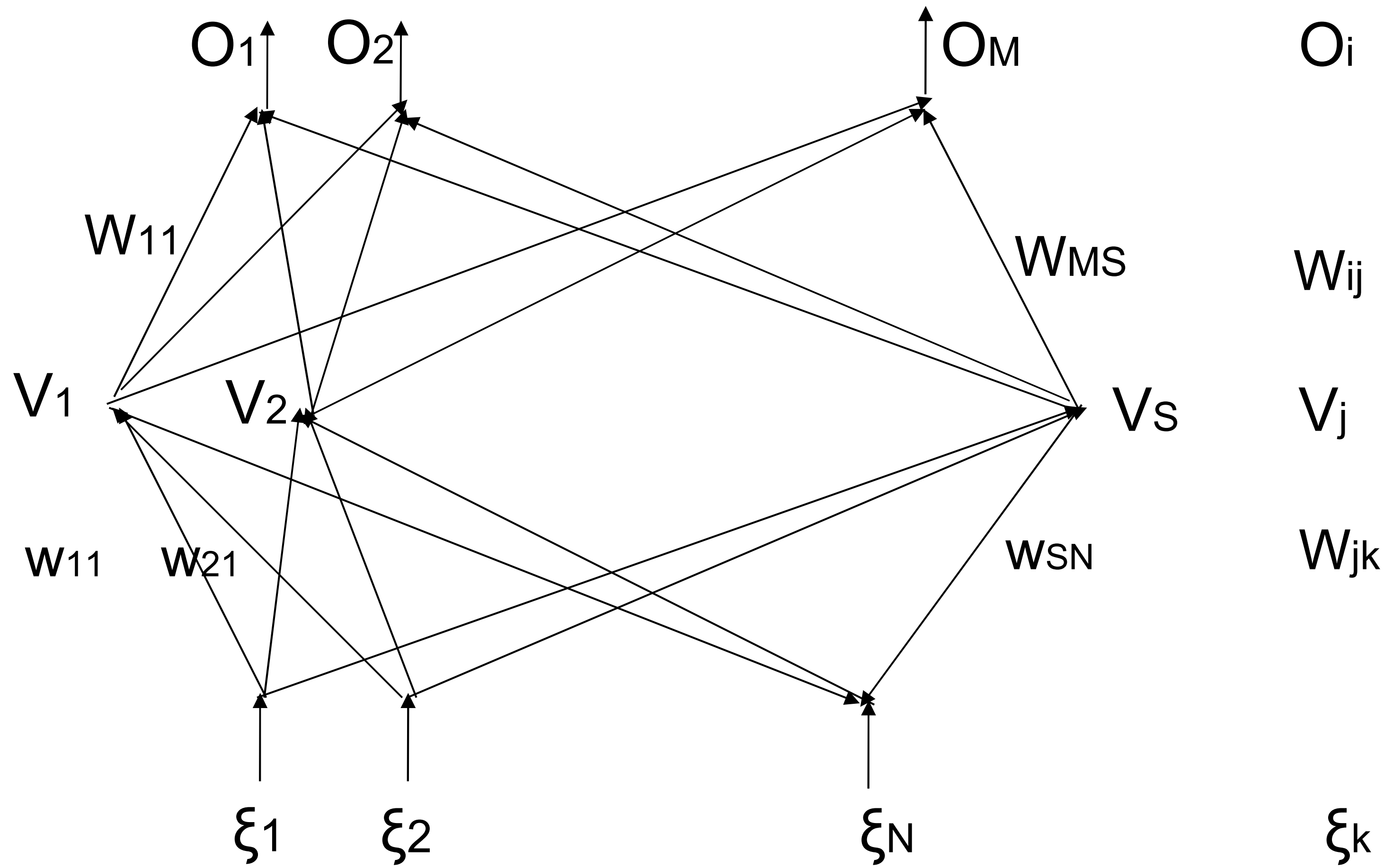
$$\tilde{f}_i(\xi_1 \ldots \xi_N) := \sum_{j=1}^{K} W_{ij}\, g\left(\sum_{k=1}^{N} w_{jk}\xi_k - \theta_j\right)$$

entonces $$\| f - \tilde{f} \|_{L^\infty(C)} = \max_{\xi \in C} | f(\xi_1 \ldots \xi_N) - \tilde{f}(\xi_1 \ldots \xi_N) | < \varepsilon$$

(vale para cualquier métrica L^p, en particular para la cuadrática)

NOTACION Y DEFINICIONES

Tomemos M=2 $\longrightarrow$ Perceptrón bicapa (una capa oculta)



Patrones: pares $(\xi(\mu),\zeta(\mu))$ $\mu=1,\ldots,p$ $\xi(\mu)=(\xi_1(\mu),\xi_2,(\mu),\ldots,\xi_N(\mu))$

$\zeta(\mu)=(\zeta_1(\mu),\zeta_2(\mu),\ldots,\zeta_M(\mu))$

Para el patrón μ como entrada, la salida será:

$$O_i^\mu = g(h_i^\mu) = g\left(\sum_{j=1}^{S} W_{ij} V_j^\mu\right) = g\left(\sum_{j=1}^{S} W_{ij}\, g\left(\sum_{k=1}^{N} w_{jk} \xi_k^\mu\right)\right) \qquad i=1,\dots,L$$

En notación matricial:

$$\vec{O}^\mu = g(W \vec{V}^\mu) = g(W g(w \vec{\xi}^\mu))$$

Función de costo o error:

$$E(w,W) = \frac{1}{2} \sum_{\mu i} \left(\zeta_i^\mu - O_i^\mu\right)^2 = \frac{1}{2} \sum_{\mu i} \left(\zeta_i^\mu - g\left(\sum_j W_{ij}\, g\left(\sum_k w_{jk} \xi_k^\mu\right)\right)\right)^2$$

cuya continuidad y diferenciabilidad dependerán de g.

Pedimos g al menos diferenciable.

EL ALGORITMO

g diferenciable $\Longrightarrow E(w)$ diferenciable $\Longrightarrow$ puede aplicarse descenso por gradiente

$$\Longrightarrow \qquad \Delta w = -\eta \nabla E \qquad (\eta \text{ velocidad})$$

Conexiones capa oculta - capa de salida:

$$\Delta W_{ij} = -\eta \frac{\partial E}{\partial W_{ij}} = \eta \sum_{\mu} \left(\zeta_i^{\mu} - O_i^{\mu} \right) g'(h_i^{\mu}) V_j^{\mu} = \eta \sum_{\mu} \delta_i^{\mu} V_j^{\mu}$$

donde $\delta_i^{\mu} := \left(\zeta_i^{\mu} - O_i^{\mu} \right) g'(h_i^{\mu})$ (1)

$\Longrightarrow$ regla δ, igual a un perceptrón simple, como si los V_j^{μ} fueran las entradas.

Conexiones entrada - capa oculta:

$$\Delta w_{jk} = -\eta \frac{\partial E}{\partial w_{jk}} = -\eta \sum_{\mu=1}^{p} \frac{\partial E}{\partial V_j^{\mu}} \frac{\partial V_j^{\mu}}{\partial w_{jk}} =$$

$$= \eta \sum_{\mu} \Big(\sum_i \underbrace{\left(\zeta_i^{\mu} - O_i^{\mu} \right) g'(h_i^{\mu})}_{\delta_i^{\mu}} W_{ij}\, g'(h_j^{\mu}) \xi_k^{\mu} \Big) = \eta \sum_{\mu i} \delta_i^{\mu} W_{ij}\, g'(h_j^{\mu}) \xi_k^{\mu} = \eta \sum_{\mu} \delta_j^{\mu} \xi_k^{\mu}$$

siendo ahora

$$\delta_j^{\mu} := g'(h_j^{\mu}) \sum_i W_{ij}\, \delta_i^{\mu} \qquad (2)$$

En general, para cualquier número de capas vale:

$$\Delta w_{pq} = \eta \sum_{\mu} \delta_{out} \cdot V_{in}$$

V_{in} entradas de la capa anterior o entradas reales

δ_{out} como en (1) o en (2) dependiendo de si es la última capa de conexiones o una anterior.

Observación: los δ de una capa oculta se calculan a partir de los de las unidades que esa capa alimenta (de ahí el nombre de error backpropagation).

IMPLEMENTACIÓN

Aprendizaje

- sincrónico: primero se calculan todas las salidas ($\forall \mu$) y luego todos los δ ($\longrightarrow$ batch)
- asincrónico: se entrena con un patrón por vez

g habituales

- $g_1(h) = f_\beta(h) = \dfrac{1}{1 + \exp(-2\beta h)}$ $\Longrightarrow$ rango $(0,1)$
- $g_2(h) = \tanh \beta h$ $\Longrightarrow$ rango $(-1,1)$

Se cumple entonces:

$$g_1'(h) = 2\beta g_1(1-g_1)$$
$$g_2'(h) = \beta(1-g_2^2)$$

$\Longrightarrow$ facilitan el cálculo de los δ_i^μ

Pasos a seguir (versión asincrónica o secuencial)

(mantenemos la notación: M número de capas
V_i^m salida de la i-ésima neurona de la m-ésima capa
$V_i^0 = \xi_i$
w_{ij}^m: conexión de V_j^{m-1} a V_i^m

1. Inicializar los w (pequeños y al azar)

2. Elegir un patrón (μ); $V_k^0 = \xi_k^\mu \quad \forall k$

3. Etapa forward: $V_i^m = g(h_i^m) = g\left(\sum_j w_{ij}^m V_j^{m-1}\right)$ $\forall i, m$ hasta los V_i^M finales

4. $\delta_i^M = g'(h_i^M)(\xi_i^\mu - V_i^M)$ deltas de la capa de salida para el patrón considerado

5. Etapa backward: retropropagación de errores

$$\delta_i^{m-1} = g'(h_i^{m-1}) \sum_j w_{ji}^m \delta_j^m \qquad m = M, M-1, \ldots, 2$$

6. $\Delta w_{ij}^m = \eta \, \delta_i^m V_j^{m-1}$

$w_{ij}^{nue} = w_{ij}^{vie} + \Delta_{ij}$

7. Ir a 2 (seleccionar patrón)

EXTENSIONES Y VARIANTES

Dos defectos principales de BP: *lento*
Mínimos locales

Algunas mejoras:

- *Momento*:

$$\Delta w_{ij}(n+1) = -\eta \partial E/\partial w_{ij} + \alpha \, \Delta w_{ij}(n)$$

--> Si superficie de costo plana, acelera en un factor $1/(1-\alpha)$
Si hay oscilaciones, las fluctuaciones son escaladas por η

- *Parámetros adaptivos*:

$$\Delta\eta = \begin{cases} +a & \text{si } \Delta E < 0 \text{ en los últimos pasos} \rightarrow \text{crece aritméticamente} \\ -\beta\eta & \text{si } \Delta E > 0 \rightarrow \text{decrece geométricamente} \\ 0 & \text{en otro caso} \end{cases}$$

Si $\Delta E > 0 \rightarrow \eta$ decrece $\rightarrow$ se anula la modificación
$a = 0$ hasta un paso exitoso (si se estaba usando momento)

- *Otras técnicas determinísticas*: steepest descent
Gradientes conjugados
Quasi-Newton

- *Técnicas estocásticas*

- <u>Gradientes conjugados</u>: $\boxed{d^{(n+1)} = -\nabla E^{(n+1)} + \beta^{(n+1)} d^{(n)}}$

Cada nueva dirección es un compromiso entre la del gradiente y la anterior.

Condición: $\beta^{(n+1)}$ tal que $d^{(n+1)}$ no cambie la componente del gradiente a lo largo de la dirección previa (componente que ya era nula):

$$d^{(n)} \cdot \nabla E\left(w^{(n+1)} + \lambda d^{(n+1)}\right) = 0$$

$$d^{(n)} \cdot \left(\nabla E\left(w^{(n+1)}\right) + H \lambda d^{(n+1)}\right) = 0$$

$$\underbrace{d^{(n)} \cdot \nabla E\left(w^{(n+1)}\right)}_{0} + \lambda d^{(n)} H d^{(n+1)} = 0$$

parado en un punto de ese segmento, la dirección del gradiente debe ser ⊥ a la del paso anterior

$\Leftrightarrow d^{(n)} H d^{(n+1)} = 0$ $\quad d^{(n)}$ y $d^{(n+1)}$ vectores conjugados

Posibles β:

- Fletcher-Reeves ('64, la original): $\boxed{\beta^{(n+1)} = \dfrac{\left(\nabla E^{(n+1)}\right)^2}{\left(\nabla E^{(n)}\right)^2}}$
- Polak-Ribière: $\boxed{\beta^{(n+1)} = \dfrac{\left(\nabla E^{(n+1)} - \nabla E^{(n)}\right)^T \cdot \nabla E^{(n+1)}}{\left(\nabla E^{(n)}\right)^2}}$

- Las últimas N (dimensión) ~~dir~~ direcciones son mutuamente conjugadas.
- No requiere conocer H.
- Superior a BP y SD en general. Encuentra el mínimo de una sup. cuadrática en N pasos.
- Computacionalmente más complejo. Sensible al λ de cada paso.