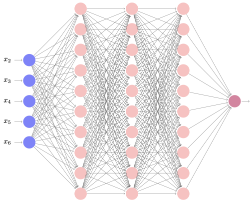

Perceptrons and Multilayer Perceptrons

The backpropagation algorithm

Vincent Barra
LIMOS, UMR 6158 CNRS, Université Clermont Auvergne

Forward pass

Train the MLP $\leftrightarrow$ Minimize a loss function ℓ over a training set Z

$$\ell = \sum_{(\boldsymbol{x},y)\in Z} \mathcal{L}\left(f_{(\boldsymbol{w},\boldsymbol{b})}(\boldsymbol{x}), y\right)$$

We use a gradient optimization process: if $(\hat{\boldsymbol{w}}, \hat{\boldsymbol{b}})$ are the optimal parameters of the MLP, then

$$\sum_{(\boldsymbol{x},y)\in Z} \nabla_{(\boldsymbol{w},\boldsymbol{b})}\mathcal{L}\left(f_{(\hat{\boldsymbol{w}},\hat{\boldsymbol{b}})}(\boldsymbol{x}), y\right) = \mathbf{0}$$

Let consider a single example $\boldsymbol{x}$, and let $s^i, i \in [\![1 \cdots L]\!]$ denote the consecutive post synaptic potential computed at each layer.

$$x \underset{\boldsymbol{w}^1,\boldsymbol{b}^1}{\longrightarrow} s^1 \underset{\sigma}{\rightarrow} x^1 \underset{\boldsymbol{w}^2,\boldsymbol{b}^2}{\longrightarrow} s^2 \underset{\sigma}{\rightarrow} \cdots \underset{\boldsymbol{w}^L,\boldsymbol{b}^L}{\longrightarrow} s^L \underset{\sigma}{\rightarrow} f_{(\boldsymbol{w},\boldsymbol{b})}(\boldsymbol{x})$$

thus (**forward pass**): $\boldsymbol{x}^{(0)} = \boldsymbol{x}$ and for $i \in [\![1 \cdots L]\!]$;

$$s^{(i)} = {\boldsymbol{w}^{(i)}}^T \boldsymbol{x}^{(i-1)} + b^{(i)} \text{ and } \boldsymbol{x}^{(i)} = \sigma(s^{(i)})$$

.

Backpropagation

The core principle of the back-propagation algorithm is the chain rule in $\mathbb{R}$:

$$(f_1 \circ f_2)' = (f_1' \circ f_2) f_2'$$

in $\mathbb{R}^d$:

$$J_{f_n \circ \dots \circ f_1}(\boldsymbol{x}) = J_{f_n}\left(f_{n-1}(\cdots(\boldsymbol{x})))\right) \cdots J_{f_3}\left(f_2(f_1(\boldsymbol{x}))\right) J_{f_2}(f_1(\boldsymbol{x})) J_{f_1}(\boldsymbol{x})$$

where $J_{f_i}(\boldsymbol{x})$ is the Jacobian of f at $\boldsymbol{x}$.

Backpropagation

Recall: $x^{(i-1)} \underset{\boldsymbol{w}^L, \boldsymbol{b}^L}{\longrightarrow} s^L \underset{\sigma}{\rightarrow} f_{(\boldsymbol{w},\boldsymbol{b})}(\boldsymbol{x})$

Then since for all components j : $x_j^{(i)} = \sigma(s_j^{(i)})$

$$\frac{\partial \mathcal{L}}{\partial s_j^{(i)}} = \frac{\partial \mathcal{L}}{\partial x_j^{(i)}} \frac{\partial x_j^{(i)}}{\partial s_j^{(i)}} = \frac{\partial \mathcal{L}}{\partial x_j^{(i)}} \sigma'(s_j^{(i)})$$

and since $x_j^{(i-1)}$ impacts $\mathcal{L}$ only through the $s_j^{(i)}$:

$$s_j^{(i)} = \sum_k w_{j,k}^{(i)} x_j^{(i-1)} + b_j^{(i)}$$

Backpropagation

Recall: $x^{(i-1)} \xrightarrow[\boldsymbol{w}^L,\boldsymbol{b}^L]{} s^L \xrightarrow[\sigma]{} f_{(\boldsymbol{w},\boldsymbol{b})}(\boldsymbol{x})$

Then since for all components j : $x_j^{(i)} = \sigma(s_j^{(i)})$

$$\frac{\partial \mathcal{L}}{\partial s_j^{(i)}} = \frac{\partial \mathcal{L}}{\partial x_j^{(i)}} \frac{\partial x_j^{(i)}}{\partial s_j^{(i)}} = \frac{\partial \mathcal{L}}{\partial x_j^{(i)}} \sigma'(s_j^{(i)})$$

and since $s_j^{(i)} = \sum_k w_{j,k}^{(i)} x_k^{(i-1)} + b_j^{(i)}$, then

$$\frac{\partial \mathcal{L}}{\partial x_k^{(i-1)}} = \sum_j \frac{\partial \mathcal{L}}{\partial s_j^{(i)}} \frac{\partial s_j^{(i)}}{\partial x_k^{(i-1)}} = \sum_j \frac{\partial \mathcal{L}}{\partial s_j^{(i)}} w_{j,k}^{(i)}$$

$$\text{and } \frac{\partial \mathcal{L}}{\partial w_{j,k}^{(i)}} = \frac{\partial \mathcal{L}}{\partial s_j^{(i)}} \frac{\partial s_j^{(i)}}{\partial w_{j,k}^{(i)}} = \frac{\partial \mathcal{L}}{\partial s_j^{(i)}} x_k^{(i-1)} \text{ and } \frac{\partial \mathcal{L}}{\partial b_j^{(i)}} = \frac{\partial \mathcal{L}}{\partial s_j^{(i)}}$$

LABORATOIRE D'INFORMATIQUE, DE MODÉLISATION ET D'OPTIMISATION DES SYSTÈMES

UNIVERSITÉ Clermont Auvergne

Backpropagation - summary

So $\frac{\partial \mathcal{L}}{\partial x_k^{(L)}}$ can be recursively computed (**backward propagated**):

$$\frac{\partial \mathcal{L}}{\partial s_j^{(l)}} = \frac{\partial \mathcal{L}}{\partial x_j^{(i)}} \sigma'(s_j^{(i)})$$

and

$$\frac{\partial \mathcal{L}}{\partial x_k^{(i-1)}} = \sum_j \frac{\partial \mathcal{L}}{\partial s_j^{(i)}} w_{j,k}^{(i)}$$

and then compute the derivatives w.r.t to $\boldsymbol{w}$ and $\boldsymbol{b}$

$$\frac{\partial \mathcal{L}}{\partial w_{j,k}^{(i)}} = \frac{\partial \mathcal{L}}{\partial s_j^{(i)}} x_k^{(i-1)} \text{ and } \frac{\partial \mathcal{L}}{\partial b_j^{(i)}} = \frac{\partial \mathcal{L}}{\partial s_j^{(i)}}$$

LIMOS LABORATOIRE D'INFORMATIQUE, DE MODÉLISATION ET D'OPTIMISATION DES SYSTÈMES

Backpropagation - Matrix formulation

Notations:

- for $f : \mathbb{R}^d \to \mathbb{R}^c$, $f = (f_1 \cdots f_c)^T$, $f_i : \mathbb{R}^d \to \mathbb{R}$

$$\left[\frac{\partial f}{\partial \boldsymbol{x}}\right] = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_d} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_c}{\partial x_1} & \cdots & \frac{\partial f_c}{\partial x_d} \end{pmatrix}$$

- for $f : \mathbb{R}^{d \times c} \to \mathbb{R}$,

$$\left[\!\left[\frac{\partial f}{\partial \boldsymbol{w}}\right]\!\right] = \begin{pmatrix} \frac{\partial f}{\partial w_{1,1}} & \cdots & \frac{\partial f}{\partial w_{1,c}} \\ \vdots & \ddots & \vdots \\ \frac{\partial f}{\partial w_{d,1}} & \cdots & \frac{\partial f}{\partial w_{d,c}} \end{pmatrix}$$

LABORATOIRE D'INFORMATIQUE,
DE MODÉLISATION ET D'OPTIMISATION DES SYSTÈMES

Backpropagation - Matrix formulation

1. Forward pass: $\boldsymbol{x}^{(0)} = \boldsymbol{x}$ and for for $i \in [\![1 \cdots L]\!]$;

$$s^{(i)} = {\boldsymbol{w}^{(i)}}^T \boldsymbol{x}^{(i-1)} + b^{(i)} \text{ and } \boldsymbol{x}^{(i)} = \sigma(s^{(i)})$$

.

2. Backward pass:

$$\begin{cases} \text{Compute ouput layer from } \mathcal{L} & \left[\frac{\partial \mathcal{L}}{\partial \boldsymbol{x}^{(L)}}\right] \\ \text{Hidden layers } i < L & \left[\frac{\partial \mathcal{L}}{\partial \boldsymbol{x}^{(i)}}\right] = {\boldsymbol{w}^{(i+1)}}^T \left[\frac{\partial \mathcal{L}}{\partial s^{(i+1)}}\right] \end{cases}$$

3. Compute $\left[\frac{\partial \mathcal{L}}{\partial \boldsymbol{s}^{(i)}}\right] = \left[\frac{\partial \mathcal{L}}{\partial \boldsymbol{x}^{(i)}}\right] \odot \sigma'(s^{(i)})$
4. Compute gradient: $\left[\!\left[\frac{\partial \mathcal{L}}{\partial \boldsymbol{w}^{(i)}}\right]\!\right] = \left[\frac{\partial \mathcal{L}}{\partial \boldsymbol{s}^{(i)}}\right] {\boldsymbol{x}^{(i-1)}}^T$ and $\left[\frac{\partial \mathcal{L}}{\partial \boldsymbol{b}^{(i)}}\right] = \left[\frac{\partial \mathcal{L}}{\partial s^{(i)}}\right]$
5. Gradient descent: $\boldsymbol{w}^{(i)} = \boldsymbol{w}^{(i)} - \eta \left[\!\left[\frac{\partial \mathcal{L}}{\partial \boldsymbol{w}^{(i)}}\right]\!\right]$ and $\boldsymbol{b}^{(i)} = \boldsymbol{b}^{(i)} - \eta \left[\frac{\partial \mathcal{L}}{\partial \boldsymbol{b}^{(i)}}\right]$

LIMOS LABORATOIRE D'INFORMATIQUE, DE MODÉLISATION ET D'OPTIMISATION DES SYSTÈMES

UNIVERSITÉ Clermont Auvergne