

## Module 4.8: Backpropagation: Pseudo code

Finally, we have all the pieces of the puzzle

Finally, we have all the pieces of the puzzle

$$\nabla_{a_L} \mathcal{L}(\theta) \quad (\text{gradient w.r.t. output layer})$$

Finally, we have all the pieces of the puzzle

$$\nabla_{a_L} \mathcal{L}(\theta) \quad (\text{gradient w.r.t. output layer})$$

$$\nabla_{h_k} \mathcal{L}(\theta), \nabla_{a_k} \mathcal{L}(\theta) \quad (\text{gradient w.r.t. hidden layers } 0 < k < L)$$

Finally, we have all the pieces of the puzzle

$$\nabla_{a_L} \mathcal{L}(\theta) \quad (\text{gradient w.r.t. output layer})$$

$$\nabla_{h_k} \mathcal{L}(\theta), \nabla_{a_k} \mathcal{L}(\theta) \quad (\text{gradient w.r.t. hidden layers } 0 < k < L)$$

$$\nabla_{W_k} \mathcal{L}(\theta), \nabla_{b_k} \mathcal{L}(\theta) \quad (\text{gradient w.r.t. weights and biases})$$

Finally, we have all the pieces of the puzzle

$$\nabla_{a_L} \mathcal{L}(\theta) \quad (\text{gradient w.r.t. output layer})$$

$$\nabla_{h_k} \mathcal{L}(\theta), \nabla_{a_k} \mathcal{L}(\theta) \quad (\text{gradient w.r.t. hidden layers } 0 < k < L)$$

$$\nabla_{W_k} \mathcal{L}(\theta), \nabla_{b_k} \mathcal{L}(\theta) \quad (\text{gradient w.r.t. weights and biases})$$

We can now write the full learning algorithm

---

**Algorithm:** `gradient_descent()`

---

$t \leftarrow 0$ ;

$max\_iterations \leftarrow 1000$ ;

*Initialize*  $\theta_0 = [W_1^0, \dots, W_L^0, b_1^0, \dots, b_L^0]$ ;

---

**Algorithm:** `gradient_descent()`

---

$t \leftarrow 0$ ;

$max\_iterations \leftarrow 1000$ ;

*Initialize*  $\theta_0 = [W_1^0, \dots, W_L^0, b_1^0, \dots, b_L^0]$ ;

**while**  $t++ < max\_iterations$  **do**

|

**end**

---



---

**Algorithm:** `gradient_descent()`

---

$t \leftarrow 0$ ;

$max\_iterations \leftarrow 1000$ ;

*Initialize*  $\theta_0 = [W_1^0, \dots, W_L^0, b_1^0, \dots, b_L^0]$ ;

**while**  $t++ < max\_iterations$  **do**

$h_1, h_2, \dots, h_{L-1}, a_1, a_2, \dots, a_L, \hat{y} = forward\_propagation(\theta_t)$ ;

**end**

---

---

**Algorithm:** `gradient_descent()`

---

 $t \leftarrow 0;$  $max\_iterations \leftarrow 1000;$  $Initialize \quad \theta_0 = [W_1^0, \dots, W_L^0, b_1^0, \dots, b_L^0];$ **while**  $t++ < max\_iterations$  **do** $h_1, h_2, \dots, h_{L-1}, a_1, a_2, \dots, a_L, \hat{y} = forward\_propagation(\theta_t);$  $\nabla \theta_t = backward\_propagation(h_1, h_2, \dots, h_{L-1}, a_1, a_2, \dots, a_L, \hat{y});$ **end**

---

---

**Algorithm:** `gradient_descent()`

---

 $t \leftarrow 0;$  $max\_iterations \leftarrow 1000;$  $Initialize \quad \theta_0 = [W_1^0, \dots, W_L^0, b_1^0, \dots, b_L^0];$ **while**  $t++ < max\_iterations$  **do** $h_1, h_2, \dots, h_{L-1}, a_1, a_2, \dots, a_L, \hat{y} = forward\_propagation(\theta_t);$  $\nabla \theta_t = backward\_propagation(h_1, h_2, \dots, h_{L-1}, a_1, a_2, \dots, a_L, \hat{y});$  $\theta_{t+1} \leftarrow \theta_t - \eta \nabla \theta_t;$ **end**

---

---

**Algorithm:** forward\_propagation( $\theta$ )

---

---

**Algorithm:** forward\_propagation( $\theta$ )

---

**for**  $k = 1$  *to*  $L - 1$  **do**

|

**end**

---

---

**Algorithm:** forward\_propagation( $\theta$ )

---

**for**  $k = 1$  *to*  $L - 1$  **do**

$a_k = b_k + W_k h_{k-1};$

**end**

---

---

**Algorithm:** forward\_propagation( $\theta$ )

---

**for**  $k = 1$  *to*  $L - 1$  **do**

$a_k = b_k + W_k h_{k-1};$   
     $h_k = g(a_k);$

**end**

---

---

**Algorithm:** forward\_propagation( $\theta$ )

---

**for**  $k = 1$  *to*  $L - 1$  **do**

$a_k = b_k + W_k h_{k-1};$   
     $h_k = g(a_k);$

**end**

$a_L = b_L + W_L h_{L-1};$

---



---

**Algorithm:** forward\_propagation( $\theta$ )

---

**for**  $k = 1$  *to*  $L - 1$  **do**

$a_k = b_k + W_k h_{k-1};$   
     $h_k = g(a_k);$

**end**

$a_L = b_L + W_L h_{L-1};$

$\hat{y} = O(a_L);$

---

Just do a forward propagation and compute all  $h_i$ 's,  $a_i$ 's and  $f(x)$

---

**Algorithm:** back\_propagation( $h_1, h_2, \dots, h_{L-1}, a_1, a_2, \dots, a_L, \hat{y}$ )

---

//Compute output gradient ;

Just do a forward propagation and compute all  $h_i$ 's,  $a_i$ 's and  $f(x)$

---

**Algorithm:** back\_propagation( $h_1, h_2, \dots, h_{L-1}, a_1, a_2, \dots, a_L, \hat{y}$ )

---

//Compute output gradient ;

$$\nabla_{a_L} \mathcal{L}(\theta) = -(e(y) - f(x)) ;$$

Just do a forward propagation and compute all  $h_i$ 's,  $a_i$ 's and  $f(x)$

---

**Algorithm:** back\_propagation( $h_1, h_2, \dots, h_{L-1}, a_1, a_2, \dots, a_L, \hat{y}$ )

---

//Compute output gradient ;

$\nabla_{a_L} \mathcal{L}(\theta) = -(e(y) - f(x))$  ;

**for**  $k = L$  *to* 1 **do**

**end**

---

Just do a forward propagation and compute all  $h_i$ 's,  $a_i$ 's and  $f(x)$

---

**Algorithm:** back\_propagation( $h_1, h_2, \dots, h_{L-1}, a_1, a_2, \dots, a_L, \hat{y}$ )

---

//Compute output gradient ;

$\nabla_{a_L} \mathcal{L}(\theta) = -(e(y) - f(x))$  ;

**for**  $k = L$  *to* 1 **do**

    // Compute gradients w.r.t. parameters ;

**end**

---

Just do a forward propagation and compute all  $h_i$ 's,  $a_i$ 's and  $f(x)$

---

**Algorithm:** back\_propagation( $h_1, h_2, \dots, h_{L-1}, a_1, a_2, \dots, a_L, \hat{y}$ )

---

//Compute output gradient ;

$$\nabla_{a_L} \mathcal{L}(\theta) = -(e(y) - f(x)) ;$$

**for**  $k = L$  *to* 1 **do**

    // Compute gradients w.r.t. parameters ;

$$\nabla_{W_k} \mathcal{L}(\theta) = \nabla_{a_k} \mathcal{L}(\theta) h_{k-1}^T ;$$

**end**

---

Just do a forward propagation and compute all  $h_i$ 's,  $a_i$ 's and  $f(x)$

---

**Algorithm:** back\_propagation( $h_1, h_2, \dots, h_{L-1}, a_1, a_2, \dots, a_L, \hat{y}$ )

---

//Compute output gradient ;

$$\nabla_{a_L} \mathcal{L}(\theta) = -(e(y) - f(x)) ;$$

**for**  $k = L$  **to** 1 **do**

    // Compute gradients w.r.t. parameters ;

$$\nabla_{W_k} \mathcal{L}(\theta) = \nabla_{a_k} \mathcal{L}(\theta) h_{k-1}^T ;$$

$$\nabla_{b_k} \mathcal{L}(\theta) = \nabla_{a_k} \mathcal{L}(\theta) ;$$

**end**

---

Just do a forward propagation and compute all  $h_i$ 's,  $a_i$ 's and  $f(x)$

---

**Algorithm:** back\_propagation( $h_1, h_2, \dots, h_{L-1}, a_1, a_2, \dots, a_L, \hat{y}$ )

---

//Compute output gradient ;

$$\nabla_{a_L} \mathcal{L}(\theta) = -(e(y) - f(x)) ;$$

**for**  $k = L$  *to* 1 **do**

    // Compute gradients w.r.t. parameters ;

$$\nabla_{W_k} \mathcal{L}(\theta) = \nabla_{a_k} \mathcal{L}(\theta) h_{k-1}^T ;$$

$$\nabla_{b_k} \mathcal{L}(\theta) = \nabla_{a_k} \mathcal{L}(\theta) ;$$

    // Compute gradients w.r.t. layer below ;

**end**

---



Just do a forward propagation and compute all  $h_i$ 's,  $a_i$ 's and  $f(x)$

---

**Algorithm:** back\_propagation( $h_1, h_2, \dots, h_{L-1}, a_1, a_2, \dots, a_L, \hat{y}$ )

---

//Compute output gradient ;

$$\nabla_{a_L} \mathcal{L}(\theta) = -(e(y) - f(x)) ;$$

**for**  $k = L$  *to* 1 **do**

    // Compute gradients w.r.t. parameters ;

$$\nabla_{W_k} \mathcal{L}(\theta) = \nabla_{a_k} \mathcal{L}(\theta) h_{k-1}^T ;$$

$$\nabla_{b_k} \mathcal{L}(\theta) = \nabla_{a_k} \mathcal{L}(\theta) ;$$

    // Compute gradients w.r.t. layer below ;

$$\nabla_{h_{k-1}} \mathcal{L}(\theta) = W_k^T (\nabla_{a_k} \mathcal{L}(\theta)) ;$$

**end**

---

Just do a forward propagation and compute all  $h_i$ 's,  $a_i$ 's and  $f(x)$

---

**Algorithm:** back\_propagation( $h_1, h_2, \dots, h_{L-1}, a_1, a_2, \dots, a_L, \hat{y}$ )

---

//Compute output gradient ;

$$\nabla_{a_L} \mathcal{L}(\theta) = -(e(y) - f(x)) ;$$

**for**  $k = L$  **to** 1 **do**

    // Compute gradients w.r.t. parameters ;

$$\nabla_{W_k} \mathcal{L}(\theta) = \nabla_{a_k} \mathcal{L}(\theta) h_{k-1}^T ;$$

$$\nabla_{b_k} \mathcal{L}(\theta) = \nabla_{a_k} \mathcal{L}(\theta) ;$$

    // Compute gradients w.r.t. layer below ;

$$\nabla_{h_{k-1}} \mathcal{L}(\theta) = W_k^T (\nabla_{a_k} \mathcal{L}(\theta)) ;$$

    // Compute gradients w.r.t. layer below (pre-activation);

**end**

---

Just do a forward propagation and compute all  $h_i$ 's,  $a_i$ 's and  $f(x)$

---

**Algorithm:** back\_propagation( $h_1, h_2, \dots, h_{L-1}, a_1, a_2, \dots, a_L, \hat{y}$ )

---

//Compute output gradient ;

$$\nabla_{a_L} \mathcal{L}(\theta) = -(e(y) - f(x)) ;$$

**for**  $k = L$  **to** 1 **do**

    // Compute gradients w.r.t. parameters ;

$$\nabla_{W_k} \mathcal{L}(\theta) = \nabla_{a_k} \mathcal{L}(\theta) h_{k-1}^T ;$$

$$\nabla_{b_k} \mathcal{L}(\theta) = \nabla_{a_k} \mathcal{L}(\theta) ;$$

    // Compute gradients w.r.t. layer below ;

$$\nabla_{h_{k-1}} \mathcal{L}(\theta) = W_k^T (\nabla_{a_k} \mathcal{L}(\theta)) ;$$

    // Compute gradients w.r.t. layer below (pre-activation);

$$\nabla_{a_{k-1}} \mathcal{L}(\theta) = \nabla_{h_{k-1}} \mathcal{L}(\theta) \odot [\dots, g'(a_{k-1,j}), \dots] ;$$

**end**

---