Back propagation -example explained

~S S Roy, 27th Sept,2025

1) Network, inputs, targets and initial weights

Architecture: 2 inputs → 2 hidden neurons → 2 outputs. Sigmoid activation everywhere.

Given (from the image):

- Inputs:
  $i_1 = 0.05,\ i_2 = 0.10$
- Targets:
  $t_{o1} = 0.01,\ t_{o2} = 0.99$
- Initial weights:

```ini

w1 = 0.15,  w2 = 0.20,  w3 = 0.25,  w4 = 0.30
w5 = 0.40,  w6 = 0.45,  w7 = 0.50,  w8 = 0.55
```

- Bias weights (used as weight * 1):

```pgsql

b1 = 0.35  (added into each hidden node net input)
b2 = 0.60  (added into each output node net input)
```

- Learning rate: $\eta = 0.5$
- Sigmoid: $\sigma(x) = \dfrac{1}{1+e^{-x}}$, derivative $\sigma'(x) = \sigma(x)(1-\sigma(x))$.

---

2) Forward pass — compute hidden and output activations

Hidden neuron h_1

Net input:

$$\begin{aligned}\text{net}_{h1} &= w_1 i_1 + w_2 i_2 + b_1 \cdot 1 \\ &= 0.15 \cdot 0.05 + 0.20 \cdot 0.10 + 0.35 \\ &= 0.0075 + 0.02 + 0.35 = 0.3775\end{aligned}$$

Activation (sigmoid):

$$\text{out}_{h1} = \sigma(0.3775) = \frac{1}{1+e^{-0.3775}} \approx 0.593269992107$$

Hidden neuron h_2

$$\begin{aligned}\text{net}_{h2} &= w_3 i_1 + w_4 i_2 + b_1 \cdot 1 \\ &= 0.25 \cdot 0.05 + 0.30 \cdot 0.10 + 0.35 \\ &= 0.0125 + 0.03 + 0.35 = 0.3925\end{aligned}$$

$$\text{out}_{h2} = \sigma(0.3925) \approx 0.596884378260$$

Output neuron o_1

$$\begin{aligned}\text{net}_{o1} &= w_5 \cdot \text{out}_{h1} + w_6 \cdot \text{out}_{h2} + b_2 \cdot 1 \\ &= 0.40 \cdot 0.593269992107 + 0.45 \cdot 0.596884378260 + 0.60 \\ &\approx 1.105905967060\end{aligned}$$

$$\text{out}_{o1} = \sigma(1.105905967060) \approx 0.751365069552$$

Output neuron o_2

$$\begin{aligned}\text{net}_{o2} &= w_7 \cdot \text{out}_{h1} + w_8 \cdot \text{out}_{h2} + b_2 \cdot 1 \\ &= 0.50 \cdot 0.593269992107 + 0.55 \cdot 0.596884378260 + 0.60 \\ &\approx 1.224921404096\end{aligned}$$

$$\text{out}_{o2} = \sigma(1.224921404096) \approx 0.772928465321$$

---

3) Compute total error (sum of squared errors)

Squared error for each output (with the $\frac{1}{2}$ factor):

$$E_{o1} = \tfrac{1}{2}(t_{o1} - \text{out}_{o1})^2 = \tfrac{1}{2}(0.01 - 0.751365069552)^2 \approx 0.274811083176$$
$$E_{o2} = \tfrac{1}{2}(t_{o2} - \text{out}_{o2})^2 = \tfrac{1}{2}(0.99 - 0.772928465321)^2 \approx 0.023560025584$$

Total error:

$$E_{\text{total}} = E_{o1} + E_{o2} \approx 0.298371108760$$

---

4) Backward pass — output layer gradients and weight updates

We compute $\dfrac{\partial E_{\text{total}}}{\partial w}$ for each output weight using chain rule.

For w_5 (connection $h_1 \to o_1$)

Chain rule:

$$\frac{\partial E_{\text{total}}}{\partial w_5} = \frac{\partial E_{\text{total}}}{\partial \text{out}_{o1}} \cdot \frac{\partial \text{out}_{o1}}{\partial \text{net}_{o1}} \cdot \frac{\partial \text{net}_{o1}}{\partial w_5}$$

Compute each term:

1. $\dfrac{\partial E_{\text{total}}}{\partial \text{out}_{o1}} = -(t_{o1} - \text{out}_{o1}) = -(0.01 - 0.751365069552) = 0.741365069552$
2. $\dfrac{\partial \text{out}_{o1}}{\partial \text{net}_{o1}} = \text{out}_{o1}(1 - \text{out}_{o1}) = 0.751365069552(1 - 0.751365069552) \approx 0.186815602212$
3. $\dfrac{\partial \text{net}_{o1}}{\partial w_5} = \text{out}_{h1} \approx 0.593269992107$

Multiply:

$$\frac{\partial E_{\text{total}}}{\partial w_5} = 0.741365069552 \cdot 0.186815602212 \cdot 0.593269992107 \approx 0.082167040564$$

Update w_5 (gradient descent, subtract $\eta\times$ gradient):

$$w_5^+ = w_5 - \eta \cdot \frac{\partial E}{\partial w_5} = 0.40 - 0.5 \cdot 0.082167040564 = 0.358916479718$$

For w_6 ($h_2 \to o_1$)

Same output delta for o_1 (use $\delta_{o1} = \frac{\partial E}{\partial \text{net}_{o1}} = 0.138498561629$ computed below).

Then:

$$\frac{\partial E}{\partial w_6} = \delta_{o1} \cdot \text{out}_{h2} = 0.138498561629 \cdot 0.596884378260 \approx 0.082667627848$$

$$w_6^+ = 0.45 - 0.5 \cdot 0.082667627848 = 0.408666186076$$

(For clarity: $\delta_{o1} = 0.741365069552 \cdot 0.186815602212 \approx 0.138498561629$.)

For output neuron o_2 (weights w_7, w_8)

Compute the output delta for o_2:

$$\frac{\partial E}{\partial \text{out}_{o2}} = -(t_{o2} - \text{out}_{o2}) = -(0.99 - 0.772928465321) = -0.217071534679$$

$$\frac{\partial \text{out}_{o2}}{\partial \text{net}_{o2}} = \text{out}_{o2}(1 - \text{out}_{o2}) = 0.772928465321(1 - 0.772928465321) \approx 0.175337034127$$

$$\delta_{o2} = \frac{\partial E}{\partial \text{net}_{o2}} = -0.217071534679 \cdot 0.175337034127 \approx -0.038098236517$$

Now gradients and updates:

$$\frac{\partial E}{\partial w_7} = \delta_{o2} \cdot \text{out}_{h1} = -0.038098236517 \cdot 0.593269992107 \approx -0.022602540477$$

$$w_7^+ = w_7 - 0.5 \cdot (-0.022602540477) = 0.50 + 0.011301270239 \approx 0.511301270239$$

$$\frac{\partial E}{\partial w_8} = \delta_{o2} \cdot \text{out}_{h2} = -0.038098236517 \cdot 0.596884378260 \approx -0.022740242216$$

$$w_8^+ = w_8 - 0.5 \cdot (-0.022740242216) = 0.55 + 0.011370121108 \approx 0.561370121108$$

Summary after output-layer updates (rounded):

```rust
w5 -> 0.358916479718
w6 -> 0.408666186076
w7 -> 0.511301270239
w8 -> 0.561370121108
```

---

5) Backward pass — hidden layer gradients and weight updates

Each hidden weight influences total error via both output neurons, so we sum contributions.

We first compute the quantity $\frac{\partial E}{\partial \text{net}_{h1}}$ (the hidden node delta).

For hidden neuron h_1

$$\frac{\partial E}{\partial \text{out}_{h1}} = \frac{\partial E_{o1}}{\partial \text{out}_{h1}} + \frac{\partial E_{o2}}{\partial \text{out}_{h1}} = \frac{\partial E_{o1}}{\partial \text{net}_{o1}} \cdot \frac{\partial \text{net}_{o1}}{\partial \text{out}_{h1}} + \frac{\partial E_{o2}}{\partial \text{net}_{o2}} \cdot \frac{\partial \text{net}_{o2}}{\partial \text{out}_{h1}}$$

We already have:

$$\frac{\partial E_{o1}}{\partial \text{net}_{o1}} = \delta_{o1} \approx 0.138498561629 \quad \text{and} \quad \frac{\partial E_{o2}}{\partial \text{net}_{o2}} = \delta_{o2} \approx -0.038098236517$$

Also $\frac{\partial \text{net}_{o1}}{\partial \text{out}_{h1}} = w_5 = 0.40,\ \frac{\partial \text{net}_{o2}}{\partial \text{out}_{h1}} = w_7 = 0.50.$

So

$$\frac{\partial E}{\partial \text{out}_{h1}} = 0.138498561629 \cdot 0.40 + (-0.038098236517) \cdot 0.50 = 0.0553994246516 - 0.0190491182585$$

Hidden derivative $\frac{\partial \text{out}_{h1}}{\partial \text{net}_{h1}} = \text{out}_{h1}(1 - \text{out}_{h1}) = 0.593269992107(1 - 0.593269992107) \approx 0.241300709.$

Thus hidden node delta:

$$\frac{\partial E}{\partial \text{net}_{h1}} = \frac{\partial E}{\partial \text{out}_{h1}} \cdot \frac{\partial \text{out}_{h1}}{\partial \text{net}_{h1}} \approx 0.0363503063931 \cdot 0.241300709 \approx 0.008771354689$$

Now gradients for incoming weights to h_1:

$$\frac{\partial E}{\partial w_1} = \frac{\partial E}{\partial \text{net}_{h1}} \cdot \frac{\partial \text{net}_{h1}}{\partial w_1} = 0.008771354689 \cdot i_1 = 0.008771354689 \cdot 0.05 \approx 0.000438567734$$

$$\frac{\partial E}{\partial w_2} = 0.008771354689 \cdot i_2 = 0.008771354689 \cdot 0.10 \approx 0.000877135469$$

Update:

$$w_1^+ = 0.15 - 0.5 \cdot 0.000438567734 \approx 0.149780716133$$

$$w_2^+ = 0.20 - 0.5 \cdot 0.000877135469 \approx 0.199561432266$$

For hidden neuron h_2

Analogous steps:

$$\frac{\partial E}{\partial \text{out}_{h2}} = \delta_{o1} \cdot w_6 + \delta_{o2} \cdot w_8 = 0.138498561629 \cdot 0.45 + (-0.038098236517) \cdot 0.55$$
$$\approx 0.062324352733 - 0.020952097084 \approx 0.041372255649$$

$$\frac{\partial \text{out}_{h2}}{\partial \text{net}_{h2}} = \text{out}_{h2}(1 - \text{out}_{h2}) = 0.596884378260(1 - 0.596884378260) \approx 0.239921460$$

Hidden delta:

$$\frac{\partial E}{\partial \text{net}_{h2}} = 0.041372255649 \cdot 0.239921460 \approx 0.009954254705$$

Gradients for weights into h_2:

$$\frac{\partial E}{\partial w_3} = 0.009954254705 \cdot i_1 = 0.009954254705 \cdot 0.05 \approx 0.000497712735$$

$$\frac{\partial E}{\partial w_4} = 0.009954254705 \cdot i_2 = 0.009954254705 \cdot 0.10 \approx 0.000995425471$$

Update:

$$w_3^+ = 0.25 - 0.5 \cdot 0.000497712735 \approx 0.249751143632$$

$$w_4^+ = 0.30 - 0.5 \cdot 0.000995425471 \approx 0.299502287265$$

---

6) Final updated weights (after one training step)

Hidden-layer weights:

```rust
```

```
w1 -> 0.149780716133
w2 -> 0.199561432266
w3 -> 0.249751143632
w4 -> 0.299502287265
```

Output-layer weights:

```rust

w5 -> 0.358916479718
w6 -> 0.408666186076
w7 -> 0.511301270239
w8 -> 0.561370121108
```

(These match the canonical worked example values.)

---

7) Notes / optional (bias updates)

- In this worked example we updated the eight weights as shown above. Bias weights b_1, b_2 could be updated in the same way because they enter their respective nets as multiplicative weights of a constant input 1.
  For example:

$$\frac{\partial E}{\partial b_2} = \delta_{o1} \cdot 1 + \delta_{o2} \cdot 1$$

  and update $b_2 \leftarrow b_2 - \eta \cdot \frac{\partial E}{\partial b_2}$.
  (If you want, I can compute those numeric bias updates and show the algebraic steps as well.)

---

8) Quick recap

- We ran one forward pass (computed hidden activations and outputs), computed the total error, then did a backward pass:
  - computed output deltas δ_{o1}, δ_{o2},
  - computed gradients w.r.t. output weights and updated them,
  - propagated errors into the hidden layer (summing contributions from both output neurons),
  - computed gradients w.r.t. hidden weights and updated them.
- Final numeric updated weights are listed above.

~S S Roy, 27th Sept,2025