
Algorithm 1 STEEPEST DESCENT WITH INACCURATE LINE SEARCH

Input: P training patterns (instances, data);

Output: Connection weights and activation levels of the units;

```
1:begin
2: Set counter  $t = 0$ ;
3: Initialize connection weights and threshold  $w_{ik}$  randomly;
4: Choose an initial learning rate,  $\alpha^{(0)}$ ;
5: repeat
6:   for  $p = 1, 2, \dots, P$  do
7:     for  $i = 1, 2, \dots, I$  do
8:        $o_{ip} = \sum_{k=0}^K w_{ik}^{(t)} x_{kp}$ ; \*  $x_{0p} = -1, w_{i0} = \theta_i$  *\
9:        $O_{ip} = g_i(o_{ip})$ ;
10:       $\delta_{ip} = -\frac{\partial E(\mathbf{w})}{\partial O_{ip}} g'_i(o_{ip})$ ;
11:    end for
12:  end for
13:  for  $i=1, 2, \dots, I$  do
14:    for  $k=0, 1, \dots, K$  do
15:       $\Delta_{ik}^{(t)} = \alpha^{(t)} \sum_{p=1}^P \delta_{ip} x_{kp}$ ; \* batch mode *\
16:       $w_{ik}^{(t+1)} = w_{ik}^{(t)} + \Delta_{ik}^{(t)}$ ;
17:    end for
18:  end for
19:   $\alpha^{(t+1)} = \eta \alpha^{(t)}$ ; \*  $0 < \eta < 1$  *\
20:   $t \leftarrow t + 1$ ,
21: until  $\alpha^{(t)} < 0$ 
22: Output  $\mathbf{w}^{(t)}$ 
23:end
```

Algorithm 2 STOCHASTIC ONLINE GRADIENT DESCENT

Input: P training patterns (instances, data);

Output: Connection weights and activation levels of the units;

```
1: begin
2:   Set counter  $t = 0$ ;
3:   Initialize connection weights and threshold  $w_{ik}$  randomly;
4:   Choose an initial learning rate,  $\alpha^{(0)}$ ;
5:   repeat
6:     Put patterns in random order;
7:     for  $p = 1, 2, \dots, P$  do
8:       for  $i = 1, 2, \dots, I$  do
9:          $o_{ip} = \sum_{k=0}^K w_{ik}^{(t)} x_{kp}$ ; \*  $x_{0p} = -1, w_{i0} = \theta_i$  *\
10:         $O_{ip} = g_i(o_{ip})$ ;
11:         $\delta_{ip} = -\frac{\partial E(\mathbf{w})}{\partial O_{ip}} g'_i(o_{ip})$ ;
12:      end for
13:      for  $k=0, 1, \dots, K$  do
14:        for  $i=1, 2, \dots, I$  do
15:           $\Delta_{ikp}^{(t)} = \alpha^{(t)} \delta_{ip} x_{kp}$ ;
16:           $w_{ik}^{(t+1)} = w_{ik}^{(t)} + \Delta_{ikp}^{(t)}$ ; \* on-line mode *\
17:        end for
18:      end for
19:    end for
20:     $\alpha^{(t+1)} = \eta \alpha^{(t)}$ ; \*  $0 < \eta < 1$  *\
21:     $t \leftarrow t + 1$ ;
22:  until  $\alpha^{(t)} < 0$ ;
23:  Output  $\mathbf{w}_i^{(t)}$   $i = 1, 2, \dots, I$ ;
24: end
```

Algorithm 3 STEEPEST DESCENT WITH INACCURATE LINE SEARCH

Input: P training patterns (instances, data);

Output: Connection weights $\mathbf{w}^*, \mathbf{W}^*$, training error $E(\mathbf{w}^*, \mathbf{W}^*)$;

```
1:begin
2: Set counter  $t = 0$ ;
3: Initialize the weights and threshold  $w_{jk}^{(0)}$  and  $W_{ij}^{(0)}$  randomly;
4: Choose an initial learning rate,  $\alpha^{(0)}$ ;
5: repeat
6:   for  $p=1,2,\dots,P$  do
7:     \* from input to hidden *\
8:     for  $j=1,2,\dots,J$  do
9:        $h_{jp} = \sum_{k=0}^K w_{jk}^{(t)} x_{kp}$ ; \*  $x_{0p} = -1$  *\
10:       $H_{jp} = \chi_j(h_{jp})$ ;
11:    end for
12:    \* from hidden to output *\
13:    for  $i=1,2,\dots,I$  do
14:       $o_{ip} = \sum_{j=0}^J W_{ij}^{(t)} H_{jp}$ ; \*  $H_{0p} = -1$  *\
15:       $O_{ip} = \omega_i(o_{ip})$ ;
16:    end for
17:    for  $i=1,2,\dots,I$  do
18:       $\delta_{O_{ip}} = -\frac{\partial E(\mathbf{w})}{\partial O_{ip}} w'_i(o_{ip})$ ;
19:    end for
20:    for  $j=1,2,\dots,J$  do
21:       $\delta_{H_{jp}} = \chi'_j(h_{jp}) \sum_{i=1}^I W_{ij}^{(t)} \delta_{O_{ip}}$ ;
22:    end for
23:  end for
24:  \* from output to hidden *\
25:  for  $i=1,2,\dots,I$  do
26:    for  $j=0,1,\dots,J$  do
27:       $\Delta W_{ij}^{(t)} = \alpha^{(t)} \sum_{p=1}^P \delta_{O_{ip}} H_{jp}$ ; \* batch mode *\
28:       $W_{ij}^{(t+1)} = W_{ij}^{(t)} + \Delta W_{ij}^{(t)}$ ;
29:    end for
30:  end for
31:  \* from hidden to input *\
32:  for  $j=1,2,\dots,J$  do
33:    for  $k=0,1,\dots,K$  do
34:       $\Delta w_{jk}^{(t)} = \alpha^{(t)} \sum_{p=1}^P \delta_{H_{jp}} x_{kp}$ ; \* batch mode *\
35:       $w_{jk}^{(t+1)} = w_{jk}^{(t)} + \Delta w_{jk}^{(t)}$ ;
36:    end for
37:  end for
38:   $\alpha^{(t+1)} = \eta \alpha^{(t)}$ ; \*  $0 < \eta < 1$  *\
39:   $t \leftarrow t + 1$ ;
40: until  $\alpha^{(t)} < 0$ ;
41: Output  $\mathbf{w}^{(t)}, \mathbf{W}^{(t)}$  and  $E(\mathbf{w}^{(t)}, \mathbf{W}^{(t)})$ ;
42:end
```

Algorithm 4 BACKPROPAGATION

Input: P training patterns (instances, data);

Output: Connection weights $\mathbf{w}^*, \mathbf{W}^*$, training error $E(\mathbf{w}^*, \mathbf{W}^*)$;

```
1:begin
2: Set counter  $t = 0$ ;
3: Initialize the weights and threshold  $w_{jk}^{(0)}$  and  $W_{ij}^{(0)}$  randomly;
4: Choose an initial learning rate,  $\alpha^{(0)}$ ;
5: repeat
6:   Put patterns in random order;
7:   for  $p=1,2,\dots,P$  do
8:     \* from input to hidden *\
9:     for  $j=1,2,\dots,J$  do
10:       $h_{jp} = \sum_{k=0}^K w_{jk}^{(t)} x_{kp}$ ; \*  $x_{0p} = -1$  *\
11:       $H_{jp} = \chi_j(h_{jp})$ ;
12:    end for
13:    \* from hidden to output *\
14:    for  $i=1,2,\dots,I$  do
15:       $o_{ip} = \sum_{j=0}^J W_{ij}^{(t)} H_{jp}$ ; \*  $H_{0p} = -1$  *\
16:       $O_{ip} = \omega_i(o_{ip})$ ;
17:    end for
18:    \* from output to hidden *\
19:    for  $i=1,2,\dots,I$  do
20:       $\delta_{O_{ip}} = \delta_{ip} = -\frac{\partial E(\mathbf{w})}{\partial O_{ip}} w'_i(o_{ip})$ ;
21:    end for
22:    \* from hidden to input *\
23:    for  $j=1,2,\dots,J$  do
24:       $\delta_{H_{jp}} = \chi'_j(h_{jp}) \sum_{i=1}^I W_{ij}^{(t)} \delta_{O_{ip}}$ ;
25:    end for
26:    \* from output to hidden *\
27:    for  $i=1,2,\dots,I$  do
28:      for  $j=0,1,\dots,J$  do
29:         $\Delta W_{ijp}^{(t)} = \alpha^{(t)} \delta_{O_{ip}} H_{jp}$ ; \* on-line mode *\
30:         $W_{ij}^{(t+1)} = W_{ij}^{(t)} + \Delta W_{ijp}^{(t)}$ ;
31:      end for
32:    end for
33:    \* from hidden to input *\
34:    for  $j=1,2,\dots,J$  do
35:      for  $k=0,1,\dots,K$  do
36:         $\Delta w_{jkp}^{(t)} = \alpha^{(t)} \delta_{H_{jp}} x_{kp}$ ; \* on-line mode *\
37:         $w_{jk}^{(t+1)} = w_{jk}^{(t)} + \Delta w_{jkp}^{(t)}$ ;
38:      end for
39:    end for
40:  end for
41:   $\alpha^{(t+1)} = \eta \alpha^{(t)}$ ;  $0 < \eta < 1$ *
42:   $t \leftarrow t + 1$ ;
43: until  $\alpha^{(t)} < 0$ ;
44: Output  $\mathbf{w}^{(t)}, \mathbf{W}^{(t)}$  and  $E(\mathbf{w}^{(t)}, \mathbf{W}^{(t)})$ ;
45:end
```

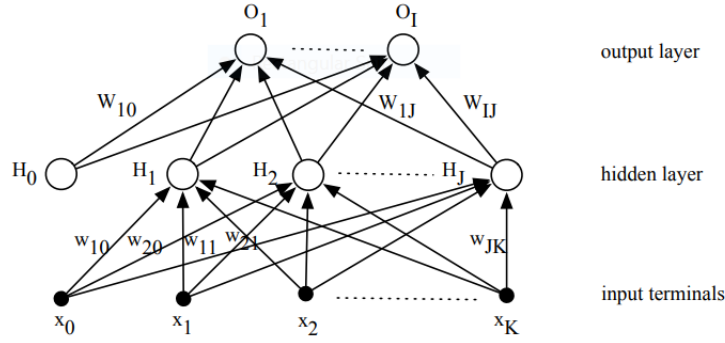


Figure 1: A two-layer perceptron

Table 1: Definitions for a two-layer perceptron

Symbol	Definition
O_i	output unit i
H_j	hidden unit j
x_k	input terminal k
W_{ij}	weight for the connection between hidden unit j and output unit i
w_{jk}	weight for the connection between input terminal k and hidden unit j
i	index for the output units, $i = 1, 2, \dots, I$
j	index for the hidden units, $j = 0, 1, 2, \dots, J$
k	index for the input terminals, $k = 0, 1, 2, \dots, K$
p	index for the input patterns, $p = 1, 2, \dots, P$
I	the number of output units
J	the number of hidden units
K	the number of input terminals
y_{ip}	target output of output unit i for pattern p
O_{ip}	actual output of output unit i for pattern p
o_{ip}	net input of output unit i for pattern p
H_{jp}	actual output of hidden unit j for pattern p
h_{jp}	net input of hidden unit j for pattern p
x_{kp}	input value of input terminal k for pattern p
$\omega_i(o)$	activation function for output unit i
$\chi_j(h)$	activation function for hidden unit j

Algorithm 5 HIDDEN_UNIT

Input: Test patterns (instances, data);

Output: The number of hidden units;

1:**begin**

2: Set counter $q = 1$;

3: Determine an initial number of hidden units, $J^{(q)}$;

4: Set $\overline{E}_T^{(0)}$ to a very large number;

5: **repeat**

6: Run backpropagation algorithm on the training data with $J^{(q)}$;

7: Find average test error $\overline{E}_T^{(q)}$ and variance $(s_T^{(q)})^2$ on the test set;

8: Report $\overline{E}_T^{(q)}$ and $(\overline{s}_T^{(q)})^2$;

9: Set $J^{(q+1)} \leftarrow J^{(q)} + 1$;

10: $q \leftarrow q + 1$;

11: **until** $\overline{E}_T^{(q)} \geq \overline{E}_T^{(q-1)}$;

12: Output $J^{(q-1)}$ as the best number of hidden units;

13:**end**
